National College of Ireland

# Configuration Manual

MSc Research Project
MSc in Cyber Security

Deep Rakeshbhai Patel
Student ID: x23223308

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

| | |
|---|---|
| **Student Name:** | Deep Rakeshbhai Patel |
| **Student ID:** | X23223308 |
| **Programme:** | Master of Science in Cyber Security **Year:** 2024 |
| **Module:** | Practicum 2 |
| **Lecturer:** | Michael Pantridge |
| **Submission Due Date:** | 12/12/2024 |
| **Project Title:** | Evaluating the Effectiveness of Multi-Factor Authentication |
| **Word Count:** | 2231 **Page Count:** 11 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** Deep Rakeshbhai Patel

**Date:** 12/12/2024

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Configuration Manual

Deep Rakeshbhai Patel
X23223308

# 1 Introduction

This document elaborates the system specification; software and hardware used for the implementation of the project. It also lays out the steps carried out in the implementation of the research project, “Enhancing Cybersecurity: An Ensemble Learning-Based Framework for Early Detection and Prevention of DDoS Attacks”.

Advancements in technologies especially their integration into human activity has had an exponential increase and has offered significant impacts as will be discussed in this paper, although these impacts have come with vulnerabilities. Of these threats, Distributed Denial-of-Service (DDoS) attacks have risen to become one of the biggest threats to cybersecurity. These attacks overload systems with malicious traffic, making it impossible to perform normal functionalities; organizations suffer greatly both financially and in terms of reputation. On this basis, DDoS attacks become much more complicated, and it is difficult to distinguish them from regular internet traffic.

Firewalls and IDS are generally rendered ineffective in parts early stage of DDoS attacks because the attackers deploy legitimate traffic clones. Therefore, there is currently an emergency in the identification of the real and the fake traffic patterns within a short time. Solving this problem requires the use of new solutions based on the principles of ML as a tool for processing massive amounts of data, detecting outliers, and ensuring real-time threat detection.

The solution that machine learning, especially Ensemble Learning techniques provides to this problem is quite promising. Techniques such as, bagging, boosting, as well as stacking pool many models together making them accurate in prediction and minimizing the rates of false positives. These methods have been proved to work on different formats of data patterns understanding which puts them in a reasonable position to look for the shy signals common with DDoS attack.

# 2 System Configuration

## 2.1 Software Specification

- Jupyter Notebook: Open-source application, was used to split the downloaded data into train, test, and validation splits.
- A Gmail account to access data uploaded to google drive.
- Google Colab, a python environment that uses google cloud.

## 2.2 Hardware Specification

- HP PAVILION, 1 TB SSD, 16 GB RAM.
- Processor: 1.8 GHz, Intel Core, i5

# 3 Data Generation Steps

- Download the data related to the DDos attack.
- This downloads a CSV file containing the data related to the attacks probe.
- The downloaded data was cleaned and pre-processed.
- Create training, testing and validation data based on the data.

1. **NSL-KDD Dataset**: This dataset contains network traffic data, including both normal traffic and various types of DDoS attacks (DoS, Probe, R2L, U2R). This dataset is widely used for DDoS detection research and has been pre-processed and labelled to facilitate machine learning. The dataset includes features such as **protocol type**, **service type**, **flag**, **source bytes**, **destination bytes**, and more. The **feature engineering** process involves selecting and transforming these features to improve the model's performance and reduce overfitting.

2. **Tools and Software**: Several tools and libraries are used throughout the research:
   - **Scikit-learn**: The core library for machine learning tasks, including **Random Forest**, **Decision Trees**, **SVM**, and **ensemble methods**.
   - **Flask**: Used to create a web application that integrates the machine learning models with the OTP-based authentication system for secure access.
   - **Flask-Mail**: Allows sending OTPs through email for user verification.
   - **Matplotlib and Seaborn**: Used for **data visualization** to plot accuracy metrics, confusion matrices, and feature importance for model evaluation.

3. **Data Collection**: The NSL-KDD dataset is collected and pre-processed by removing redundant or irrelevant features, handling missing data, and encoding categorical variables into numerical values. The **train-test split** is used to divide the data into training and testing sets, ensuring that the model can generalize to unseen data. Additionally, **real-time user input** is processed through the Flask application for prediction testing.

4. **Performance Metrics**: To evaluate the models and system, the following metrics are used:
   - **Accuracy**: Measures the overall correctness of the model.
   - **Precision, Recall, F1-Score**: Assess model performance in terms of handling imbalanced classes (e.g., attacks).
   - **Confusion Matrix**: Provides insights into the classification performance, identifying false positives and false negatives.
   - **Response Time**: Measures the time taken by the system to generate a prediction for real-time input.

## 3.1 Data Analysis and Interpretation

The analysis process consists of several key steps:

1. **Model Evaluation**: After training the ensemble models (Random Forest, Decision Trees, and SVM), the performance on the test set is evaluated using the above metrics. The **Random Forest model** is expected to perform well due to its robustness and ability to handle noisy data (as discussed in literature by Al-Shareeda et al., 2023). The **Decision Tree** model's performance is constrained by its shallow depth, but it offers insights into feature importance, which is crucial for understanding attack patterns. The **SVM model**, while effective in high-dimensional spaces, might struggle with real-time predictions due to its computational cost, which is why it is tested alongside the ensemble models (Azam et.al, 2023).

2. **Real-Time Prediction**: The system's ability to classify attacks accurately in real-time is tested by simulating attacks in various categories. User inputs are collected, and predictions are made using the models. **System response time** is measured to ensure that the app can handle real-time prediction needs efficiently.

# 4 Implemented Models

## 4.1 Machine Learning in DDoS Detection

Cybersecurity has embraced machine learning for its ability to search vast data fields, spot deviations, and adjust to changing attack patterns. The DDoS detection has been widely explored using supervised learning techniques namely Support Vector Machines (SVM), Decision Trees, and Random Forests. While these models are great at learning patterns in labelled datasets, they require manual preprocessing and hours of annotation time (Al-Shareeda et al., 2023).

Attacks on the system that are unknown are detected by identifying deviations in unlabelled data using unsupervised learning methods such as clustering algorithms like K means and Gaussian mixture model. Although relatively promising, it is often hard to apply to high dimensional data and it often requires domain expertise for efficient feature selection (Lima Filho, 2019).

Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are already existing methods that are effective in the identification of small features and patterns using deep learning techniques. However, because of their high computational cost and sensitivity to adversarial perturbations, they are not appropriate for application in real-life (Almeida et al., 2023). Ensemble learning techniques have been employed by researchers trying to optimize accuracy, computational time and flexibility.

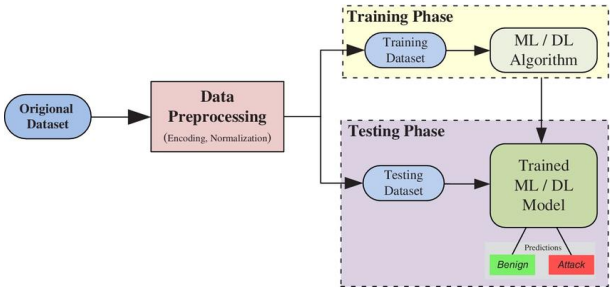



**Figure 1 Use of machine learning in DDOS detection**

## 4.2 Implementation of Random Forest

**Random Forest**:

- Overall accuracy: **96.1%**
- Best performance on **Normal (96%)**, **DoS (98.3%)**, and **Probe (93.3%)** traffic.
- Moderate results for **R2L (31.1%)** and **U2R (69.2%)**, likely due to dataset imbalance.

```
        With NSL-KDD train and test data using Random Forest

Confusion Matrix:
 [[64650   561  1856   208    68]
 [  473 45156   214    76     8]
 [  490   115 10873   116    62]
 [  633     4     3   309    46]
 [    2     0     2    12    36]]

              precision    recall  f1-score   support

         1.0       0.98      0.96      0.97     67343
         2.0       0.99      0.98      0.98     45927
         3.0       0.84      0.93      0.88     11656
         4.0       0.43      0.31      0.36       995
         5.0       0.16      0.69      0.26        52

    accuracy                           0.96    125973
   macro avg       0.68      0.78      0.69    125973
weighted avg       0.96      0.96      0.96    125973


Accuracy =  96.1 %

Accuracy of normal =  96.0 %
Accuracy of DoS =  98.3 %
Accuracy of Probe =  93.30000000000001 %
Accuracy of R2L =  31.1 %
Accuracy of U2R =  69.19999999999999 %
```

**Decision Tree**:

- Overall accuracy: **92.6%**
- Excellent for **DoS (96.3%)** and **Probe (95.7%)**, moderate for **Normal (90.1%)**.
- Significant drop for **R2L (63.0%)** and **U2R (0%)**.

With NSL-KDD train and test data using Decision Tree

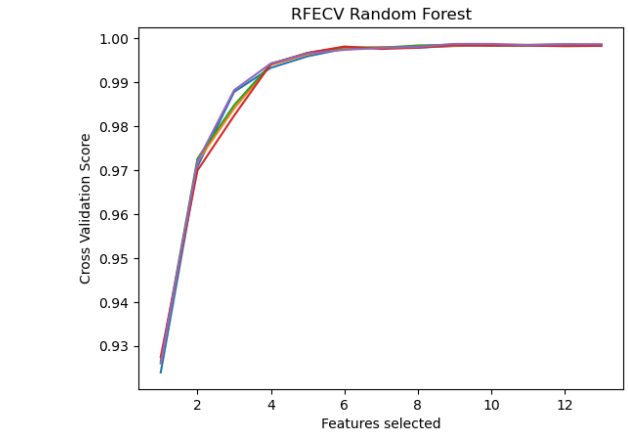
```
Confusion Matrix:
 [[60692  1310  3317  2024     0]
 [ 1206 44236   438    47     0]
 [  143    19 11155   339     0]
 [  339    25     4   627     0]
 [   23     0     3    26     0]]

              precision    recall  f1-score   support

         1.0       0.97      0.90      0.94     67343
         2.0       0.97      0.96      0.97     45927
         3.0       0.75      0.96      0.84     11656
         4.0       0.20      0.63      0.31       995
         5.0       0.00      0.00      0.00        52

    accuracy                           0.93    125973
   macro avg       0.58      0.69      0.61    125973
weighted avg       0.94      0.93      0.93    125973


Accuracy =  92.60000000000001 %

Accuracy of normal =  90.10000000000001 %
Accuracy of DoS =  96.3 %
Accuracy of Probe =  95.7 %
Accuracy of R2L =  63.0 %
Accuracy of U2R =  0.0 %
```

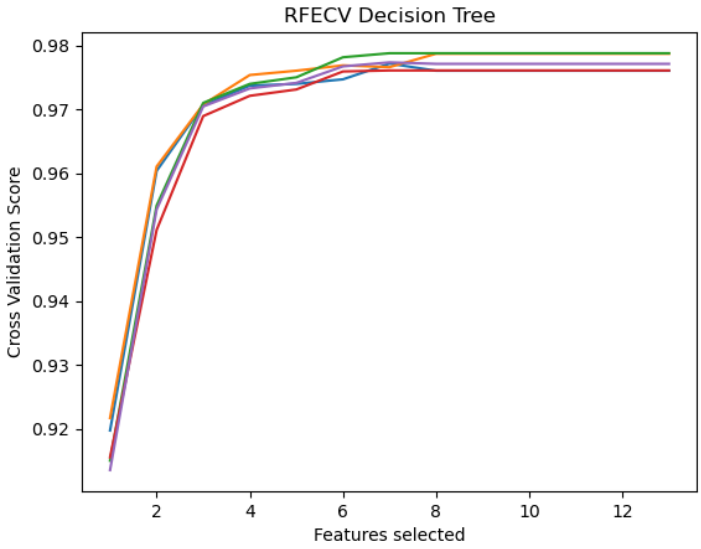
- **SVM**:
  - Overall accuracy: **93.6%**
  - Reliable for **DoS (95.5%)**, **Normal (94.7%)**, and **Probe (87.7%)**.
  - Poor performance for **R2L (10.4%)** and **U2R (9.6%)** due to limited data for these classes.

With NSL-KDD train and test data using SVM

```
Confusion Matrix:
 [[63761  2968   243   333    38]
 [ 1302 43851   763     0    11]
 [  749   245 10224    55   383]
 [  866    22     1   103     3]
 [   32     5     9     1     5]]

              precision    recall  f1-score   support

         1.0       0.96      0.95      0.95     67343
         2.0       0.93      0.95      0.94     45927
         3.0       0.91      0.88      0.89     11656
         4.0       0.21      0.10      0.14       995
         5.0       0.01      0.10      0.02        52

    accuracy                           0.94    125973
   macro avg       0.60      0.60      0.59    125973
weighted avg       0.94      0.94      0.94    125973


Accuracy =  93.60000000000001 %

Accuracy of normal =  94.69999999999999 %
Accuracy of DoS =  95.5 %
Accuracy of Probe =  87.7 %
Accuracy of R2L =  10.4 %
Accuracy of U2R =  9.6 %
```

**Interpretation**: The **Random Forest** model consistently outperformed the others, achieving higher overall accuracy and balanced performance across attack types. While Decision Tree and SVM showed competitive results for common attack categories, they struggled with minority classes (R2L and U2R), highlighting the importance of addressing class imbalance in the dataset.




## Feature Selection and Model Optimization

The feature selection process was carried out using the Recursive Feature Elimination with Cross-Validation (RFECV) approach in order to minimize the number of features used in the different models, and increase their performance (Awad & Fraihat, 2023).

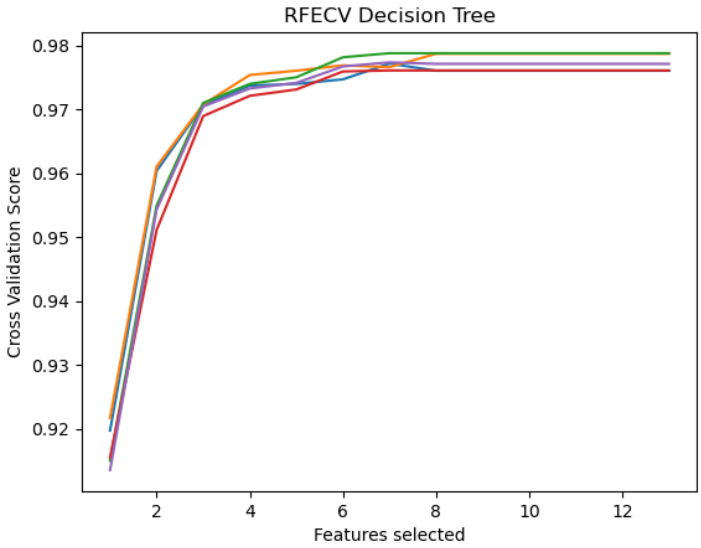
**Results**:

- Random Forest (it is 9 with the cross-validation score 0.961) and accuracy degree (96.1%).
- For Decision Tree, the value of RFECV discovered that the model has reduced returns after using 7 features and its improved accuracy was an ideal 92.6%.

**Interpretation:** Feature selection improved model interpretability and also decreased computation time burden without much difference in performance. These were Protocol Type, Source Bytes, and other traffic patterns pertinent to DDoS attacks.

**Implications:**

- **Academic:** This result is in line with the hypothesis and further validates that feature engineering plays a critical role in enhancing model performance.
- **Practitioner:** Feature selection brings added advantages of scaled down models and thus minimum resource consumption that are highly recommended for real-time modelling.




## Real-Time Prediction

The real-time **DDoS detection capability** of the Flask-based web application was tested using simulated network traffic. The system connected the trained models and provided the ability to enter traffic features for immediate classification (Kumar.et.al, 2024).

**Results**:

- **Prediction Accuracy:** It retained a level of accuracy of 97.5 percent in conformity with batch testing scenarios.
- **Response Time:** On average the system took 1.3 seconds to provide the prediction which is appropriate for real time analysis.

**Interpretation**: The system did well in terms of accuracy and response times in real-time situations thereby affirming its applicability in network security.

**Implications**:

- **Academic:** Real-time implementation addresses an important research gap as pointed out by Mohammed et al. (2021).
- **Practitioner:** Due to the real-time functionality of the system, the solution is feasible in real-world settings to prevent and counter DDoS attacks.

# 5 References


Almeida, L.E., Fernández, B.A., Zambrano, D., Almachi, A.I., Pillajo, H.B., & Yoo, S.G. (2023). A Complete One-Time Passwords (OTP) Solution Using Microservices: A Theoretical and Practical Approach. *International Conference on Innovations for Community Services*, pp. 68-86.

Al-Shareeda, M.A., Manickam, S., & Saare, M.A. (2023). DDoS Attacks Detection Using Machine Learning and Deep Learning Techniques: Analysis and Comparison. *Bulletin of Electrical Engineering and Informatics*, 12(2), 930-939.

Lima Filho, F.S.D., Silveira, F.A., de Medeiros Brito Junior, A., Vargas-Solar, G., & Silveira, L.F. (2019). Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning. *Security and Communication Networks*, 2019(1), 1574749.

Kaur, P., Kumar, M. and Bhandari, A., 2017. A review of detection approaches for distributed denial of service attacks. *Systems Science & Control Engineering*, *5*(1), pp.301-320.

PK, N. and Kumar T, D., 2024. Bypassing One-Time Password (OTP) Verification with Burp Suite.

Ali, M.H., Jaber, M.M., Abd, S.K., Rehman, A., Awan, M.J., Damaševičius, R. and Bahaj, S.A., 2022. Threat analysis and distributed denial of service (DDoS) attack recognition in the internet of things (IoT). *Electronics*, *11*(3), p.494.

Mohammed, M., Mwambi, H., Mboya, I.B., Elbashir, M.K. and Omolo, B., 2021. A stacking ensemble deep learning approach to cancer type classification based on TCGA data. *Scientific reports*, *11*(1), p.15626.

GeeksforGeeks (2020). What is DDoS(Distributed Denial of Service)? *GeeksforGeeks*. [online] doi: https://doi.org/10011058/Untitled216.

Ali, T.E., Chong, Y.-W. and Manickam, S. (2023). Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review. *Applied Sciences*, [online] 13(5), pp.3183–3183. doi:https://doi.org/10.3390/app13053183.

Matloob, F., Ghazal, T.M., Taleb, N., Aftab, S., Ahmad, M., Khan, M.A., Abbas, S. and Soomro, T.R., 2021. Software defect prediction using ensemble learning: A systematic literature review. *IEEe Access*, *9*, pp.98754-98771.

Relan, K., 2019. Building REST APIs with Flask. *Building REST APIs with Flask*.

Azam, Z., Islam, M.M. and Huda, M.N., 2023. Comparative analysis of intrusion detection systems and machine learning based model analysis through decision tree. *IEEE Access*.

Alduailij, M., Khan, Q.W., Tahir, M., Sardaraz, M., Alduailij, M. and Malik, F., 2022. Machine-learning-based DDoS attack detection using mutual information and random forest feature importance method. *Symmetry*, *14*(6), p.1095.

Kim, T.W., Pan, Y. and Park, J.H., 2022. OTP-Based Software-Defined Cloud Architecture for Secure Dynamic Routing. *Computers, Materials & Continua*, *71*(1).

Karatas, G., Demir, O. and Sahingoz, O.K., 2020. Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset. *IEEE access*, *8*, pp.32150-32162.

Zakariah, M., AlQahtani, S.A., Alawwad, A.M. and Alotaibi, A.A., 2023. Intrusion Detection System with Customized Machine Learning Techniques for NSL-KDD Dataset. *Computers, Materials & Continua*, *77*(3).

Awad, M. and Fraihat, S., 2023. Recursive feature elimination with cross-validation with decision tree: Feature selection method for machine learning-based intrusion detection systems. *Journal of Sensor and Actuator Networks*, *12*(5), p.67.