National College of Ireland

Configuration Manual

MSc Research Project
Cybersecurity

Ashwan Teja Dasari
Student ID: X23166771

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ashwan Teja Dasari
Student ID:	x23166771
Programme:	MSc in cybersecurity **Year:** 2023-2024
Module:	MSc Research Project
Lecturer:	Mr Niall Heffernan
Submission Due Date:	12-08-2024
Project Title:	Protecting User Privacy: Advanced Techniques for detecting and preventing Keyloggers Using Machine Learning

Word Count: 884 **Page Count: 8**

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: Ashwan Teja Dasari

Date: 12-08-2024

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

Ashwan Teja Dasari
Student ID: x23166771

1 Introduction

This configuration manual outlines the fundamental setup and equipment requirements of the project. It uses modern concepts and methods in machine learning to detect and mitigate threats related to keyloggers thus protecting user's privacy. This configuration manual report is important because it explains all the software and hardware implementation methods which is used deploy and manage in keylogger detecting and preventing system

2 Hardware Requirements

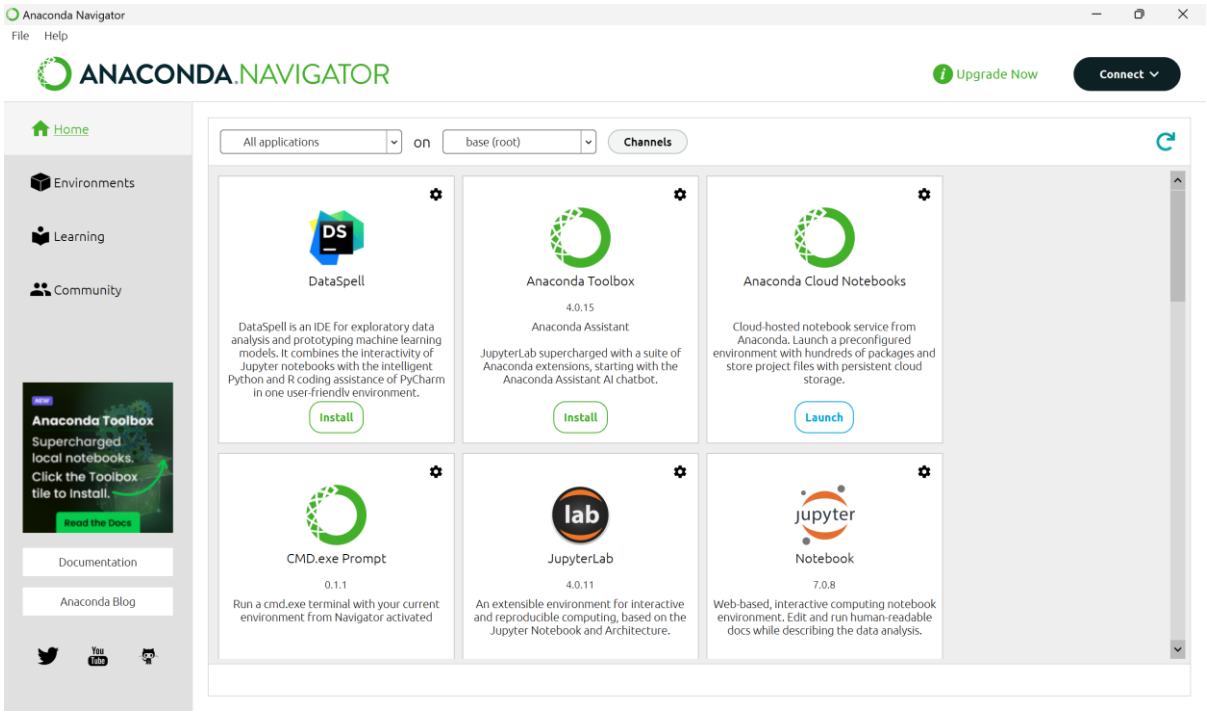
Operating System: Windows 11
RAM: 16 GB
Processor: 13th Gen Intel(R) Core (TM) i7-13700H @ 2.40 GHz
Storage: 512 GB SSD
System Type: 64-bit (OS) operating system, x64-based processor

3 Software Requirements

Jupiter Notebook
Anaconda navigator
Python 3.12.3

For this Project I have used a application called Anaconda Navigator. It comes along with Jupiter notebook, and it has inbuilt Python setup to run the code. This anaconda navigator can be installed through the below link. After installation we need to launch the Jupiter Notebook from the navigator.

Download Now | Anaconda



4 Python Libraries

Installing the libraries

```
pip install pandas numpy matplotlib seaborn scikit-learn
```

Importing the installed libbries

```
# Data manipulation and analysis
import pandas as pd
import numpy as np

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Machine learning models and tools
from sklearn.model_selection import train_test_split,
GridSearchCV,cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score, roc_auc_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier

# Ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

Pandas, NumPy, matplotlib, seaborn and scikit-learn this python libraries are used in this project.

Pandas: Python provides a unique data structure to represent a tabular data called DataFrame and pandas is a high-performance, easy-to-use library for data analysis based on this structure. It introduces the concepts of Series and DataFrame that are used to handle the

labeled data. Pandas is also efficient on handling data, be it importing from different file formats, cleaning, transforming and analysing of data.

NumPy: It has multidimensional array objects with high execution speed and includes many high-level arithmetic operations on these arrays. And also many numerical Python packages start from NumPy as it works with well-proven data structures and numerical algorithms.

Matplotlib: Matplotlib is a flexible plotting library in Python which can be used to plot static, animated as well as interactive plots. This type provides various kinds of plot available and some of the features are as follows: Matplotlib is a good platform for plotting and analysing patterns and trends in data.

Seaborn: Seaborn is an API of visualization tools designed on top of Matplotlib. It is a good tool to use when you want to develop a nice looking, informative and appealing statistical data chart. Heat maps, scatter plots as well as time series are difficult to make in matplotlib, but Seaborn provides a more convenient way of creating them that's why Seaborn is preferred for exploratory data analysis and report generation.

Scikit-learn: The scikit-learn is a machine learning library that has NumPy, SciPy and Matplotlib as basic libraries. It also offers some forms of both supervised and unsupervised learning algorithms like classification, regression, clustering, and model assessment. Simplicity and effectiveness of using the objects make scikit-learn the favorite of data scholars and ML practitioners to build and benchmark the machine learning models proficiently.

5 Data preprocessing

Importing Data:

```
data = pd.read_csv("D:\\NCi\\t\\Keylogger_Detection.csv")
```

Data Information:

- Data Information

```
data.info()
```

Data Size:

- Data Size

```
data.shape
```

```
(523617, 86)
```

3.1 Data Cleaning

```
# Fill missing values with forward fill method
data = data.fillna(method='ffill')


# 2. Remove duplicate rows
data = data.drop_duplicates()
```

3.2 Feature Engineering

```
# This can be based on the sum of packet sizes or other relevant columns
data['Total_Data_Transferred'] = data['Total Length of Fwd Packets'] + data['Total Length of Bwd Packets']

#  Normalize or scale features if necessary ( Min-Max scaling)
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
data[['Total Length of Fwd Packets',
      'Total Length of Bwd Packets',
      'Total_Data_Transferred']] = scaler.fit_transform(data[['Total Length of Fwd Packets',
                                                              'Total Length of Bwd Packets',
                                                              'Total_Data_Transferred']])


# Save the cleaned and prepared dataset to a new CSV file
data.to_csv('cleaned_network_traffic_data.csv')
```

6 Modelling & Testing

Testing has been done by using Random Forest, decision tree and Gradient Boost classifiers

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.2,
                                                    random_state=42
                                                    )
# Drop specified columns
X_train = X_train.drop(columns=columns_to_drop, errors='ignore')

# Check for non-numeric columns
non_numeric_cols = X_train.select_dtypes(include=['object']).columns
if len(non_numeric_cols) > 0:
    print("Non-numeric columns found: ", non_numeric_cols)
    # Convert non-numeric columns to numeric (e.g., using one-hot encoding)
    X_train = pd.get_dummies(X_train, columns=non_numeric_cols)
```

Here, I'm evaluating three different types of algorithms to find the best algorithm which can detect keyloggers more significantly.

```
# Define the models to evaluate
models = {

    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Gradient Boosting': GradientBoostingClassifier(),

}
```

```
# Evaluate each model using cross-validation
for model_name, model in models.items():
    print(f"Evaluating {model_name}...")
    scores = cross_val_score(model, X_train, y_train, cv=5)
    print(f"Cross-validation scores: {scores}")
    print(f"Mean cross-validation score: {np.mean(scores)}\n")
```

By observing below evaluation Random Forest has the best results.

```
Evaluating Decision Tree...
Cross-validation scores: [0.93510307 0.93867198 0.93833777 0.93574685 0.93868319]
Mean cross-validation score: 0.9373085723869735

Evaluating Random Forest...
Cross-validation scores: [0.94914    0.94799413 0.9488416  0.94873356 0.94885292]
Mean cross-validation score: 0.9487124400780113

Evaluating Gradient Boosting...
Cross-validation scores: [0.64813378 0.65096265 0.65111782 0.64785505 0.64898899]
Mean cross-validation score: 0.6494116588740382
```

7 Evaluation

```
# Train and evaluate each model on the test set


X_test = X_test.drop(columns=columns_to_drop, errors='ignore')
for model_name, model in models.items():
    print(f"Training and evaluating {model_name}...")
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
    print(f"Confusion Matrix:\n{confusion_matrix(y_test, y_pred)}")
    print(f"Classification Report:\n{classification_report(y_test, y_pred)}\n")
```

Decision Tree:

Classification Report:

	precision	recall	f1-score	support
Benign	0.97	0.96	0.96	61757
Keylogger	0.95	0.95	0.95	42967
accuracy			0.96	104724
macro avg	0.96	0.96	0.96	104724
weighted avg	0.96	0.96	0.96	104724

Random Forest:

Classification Report:

	precision	recall	f1-score	support
Benign	0.96	0.98	0.97	61757
Keylogger	0.97	0.95	0.96	42967
accuracy			0.97	104724
macro avg	0.97	0.96	0.96	104724
weighted avg	0.97	0.97	0.97	104724

Gradient Boost:

Classification Report:

	precision	recall	f1-score	support
Benign	0.64	0.93	0.76	61757
Keylogger	0.71	0.24	0.36	42967
accuracy			0.65	104724
macro avg	0.67	0.58	0.56	104724
weighted avg	0.67	0.65	0.59	104724

8 References

Anaconda. (n.d.). Retrieved may 25, 2024, from https://www.anaconda.com/download/success

Gupta, T. R. (2020, May 07). *A Survey on Machine Learning Approaches and Its Technique*. (2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS), Bhopal, India, 2020, pp. 1-6,) Retrieved june 23, 2024, from IEEE Explore: https://doi.org/10.1109/SCEECS48394.2020.190

Installing packages. (n.d.). Retrieved july 20, 2024, from https://packaging.python.org/en/latest/tutorials/installing-packages/

Keylogger detection. (n.d.). Retrieved May 22, 2024, from Kaggle: https://www.kaggle.com/datasets/subhajournal/keylogger-detection

Machine learning algorithms. (n.d.). Retrieved may 16, 2024, from https://www.simplilearn.com/10-algorithms-machine-learning-engineers-need-to-know-article

matplotlib. (n.d.). Retrieved july 20, 2024, from https://matplotlib.org/stable/install/index.html

scikit learn. (n.d.). Retrieved july 20, 2024, from https://scikit-learn.org/stable/