National College of Ireland

# Configuration Manual

MSc Research Project
Cloud Computing

Albin Biju
Student ID: 23286199

School of Computing
National College of Ireland

Supervisor: Sean Heeney

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

**Student Name:** ……Albin Biju……

**Student ID:** ……23286199……

**Programme:** ……Cloud Computing…… **Year:** …2024-2025……..

**Module:** ……MSc Research Project……

**Lecturer:** ……Sean Heeney……

**Submission Due Date:** ……11-08-2025……

**Project Title:** Improved Machine Learning-Based Intrusion Detection Systems for Secure Network Virtualization

**Word Count:** ……1288…… **Page Count:** ……4……

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:** ……Albin Biju……

**Date:** ……11-08-2025……

**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

Albin Biju
Student ID: 23286199

# 1 Introduction

This document provides a comprehensive, step-by-step guide for configuring and deploying the "Improved Machine Learning-Based Intrusion Detection System for Secure Network Virtualization." The system is designed as an end-to-end solution that combines a high-accuracy machine learning model with a robust, scalable cloud architecture built on Amazon Web Services (AWS).

This manual is intended for system administrators, cybersecurity analysts, and researchers who wish to replicate, test, or deploy the system. By following the instructions outlined herein, you will be able to:

- Set up the required local Python environment.
- Prepare the machine learning models using the provided notebooks.
- Programmatically provision the necessary AWS infrastructure, including IAM roles, S3 buckets, Lambda functions, and CloudWatch monitoring components.
- Configure and run the real-time IDS engine and its associated web dashboard.
- Verify that all components are functioning correctly.

Upon completion, you will have a fully operational intrusion detection system capable of real-time analysis, serverless inference, cloud-based monitoring, and standardized security logging.

# 2 Prerequisites

Before beginning the configuration process, please ensure you have the following prerequisites in place.

## 2.1 Software and Hardware Requirements

- **Operating System:** A modern Windows, macOS, or Linux operating system.
- **Python:** Python version 3.11 or later.
- **Pip:** Python's package installer, which typically comes with Python.
- **Project Files:** A complete, unzipped copy of the project source code.
- **Web Browser:** A modern web browser such as Chrome, Firefox, or Edge.

## 2.2 AWS Account and Tools

- **AWS Account:** An active AWS account with administrative privileges to create IAM roles, S3 buckets, Lambda functions, and CloudWatch resources.
- **AWS CLI:** The AWS Command Line Interface must be installed on your local machine.
- **AWS Credentials:** You must have the access key ID and secret access key for an IAM user with administrative permissions. For this project, the provided AdminUser_accessKeys.csv can be used.

# 3 Configuration and Deployment Steps

This section details the sequential steps required to deploy the entire system. Please follow them in the order presented.

# 4 Step 1: Configure AWS CLI

The first step is to configure your local AWS CLI to communicate with your AWS account.

1. Open a terminal or command prompt.
2. Run the following command:

aws configure

3. The CLI will prompt you for four pieces of information. Use the credentials from the AdminUser_accessKeys.csv file and the specified region:
    - **AWS Access Key ID:** Enter the Access Key ID (e.g., AKIAWQ46BJFWJ3ASUL5G).
    - **AWS Secret Access Key:** Enter the Secret Access Key (e.g., g30xKCZVaIPsas7m2rJ6Gd/J2an394pOQyJScDJC).
    - **Default region name:** Enter eu-west-1.
    - **Default output format:** Press Enter to accept the default (json).

# 5 Step 2: Set Up the Python Environment

It is highly recommended to use a virtual environment to manage project dependencies.

1. Navigate to the root directory of the project in your terminal.
2. Create a virtual environment:

python -m venv venv

3. Activate the virtual environment:
    - **On Windows:**

.\venv\Scripts\activate

    - **On macOS/Linux:**

source venv/bin/activate

4. Install the required Python packages by running:

pip install flask flask-socketio psutil boto3 pandas numpy scikit-learn==1.4.2 joblib tensorflow

*Note: A specific version of scikit-learn is used to ensure compatibility with the saved model artifacts.*

# 6 Step 3: Prepare the Machine Learning Models

The real-time IDS relies on pre-trained models. You must generate these models using the provided Jupyter notebooks.

1. Launch Jupyter Notebook or JupyterLab from your terminal:

jupyter notebook

2. First, open and run all cells in balanced-dataset-generation.ipynb. This notebook will process the raw dataset files and create the balanced_dataset.csv file, which is essential for the next step.
3. After the first notebook completes, open and run all cells in modelling.ipynb. This notebook will:
    - Load the balanced dataset.
    - Train and evaluate the Random Forest, MLP, and LSTM models.
    - Perform hyperparameter tuning.
    - Save the best-performing model (best_model_tuned_random_forest.pkl) and all other necessary artifacts (standard_scaler.pkl, label_encoder.pkl, etc.) into the /models directory.

This step is complete once the /models directory is populated with .pkl and .h5 files.

# 7 Step 4: Deploy AWS Infrastructure

These steps use the provided Python scripts to provision the cloud components.

1. **Create the IAM Role for Lambda:**
    - In your terminal (with the virtual environment still active), run the following script:

python create_lambda_role.py

    - This script creates the lambda-execution-role. It will output the **ARN (Amazon Resource Name)** of the created role. **Copy this ARN**, as you will need it in the next step.
2. **Deploy the Lambda Function:**
    - Open the deploy_lambda.py file in a text editor.
    - Locate the line:

LAMBDA_ROLE_ARN = 'arn:aws:iam::448618645868:role/lambda-execution-role' # Update with your role ARN

    - **Replace the entire ARN string** with the one you copied from the previous step. Save the file.
    - Now, run the deployment script:

python deploy_lambda.py

    - This script will package the Lambda code, create the S3 buckets (albin-thesis-model-bucket-2025 and albin-thesis-security-lake-2025), upload the model artifacts, and deploy the real-time-ids-inference Lambda function.
3. **Set Up CloudWatch Monitoring:**
    - Run the monitoring setup script:

python setup_cloudwatch_monitoring.py

    - This script will create the necessary CloudWatch Log Groups, the RealTimeIDS dashboard, all required alarms, and an SNS topic for notifications.
    - During execution, it will prompt you to enter an email address for alerts. Provide your email and check your inbox for a subscription confirmation link from AWS. You must click this link to receive alerts.

At this point, your entire cloud infrastructure is configured and ready.

# 8 Running the System

The system consists of two main processes that must be run simultaneously in separate terminals.

1. **Start the Real-Time IDS Engine:**
    - Open a new terminal, navigate to the project root, and activate the virtual environment (source venv/bin/activate or .\venv\Scripts\activate).
    - Run the main IDS script:

python real_time_ids.py

    - You should see console output indicating that the IDS is starting, loading models, and beginning packet capture. Leave this terminal running.
2. **Start the Flask Web Dashboard:**
    - Open a **second** terminal, navigate to the project root, and activate the virtual environment.
    - Run the Flask application script:

python flask_app.py

    - The console will indicate that the web server is running on http://0.0.0.0:5000.

# 9 System Verification

To verify that the system is operating correctly, follow these checks:

1. **Access the Web Dashboard:** Open your web browser and navigate to http://127.0.0.1:5000. You should see the main dashboard with live updates for threat detections and network statistics.
2. **Check Console Output:** Look at the terminal running real_time_ids.py. You should see log messages for "Normal traffic detected" and, if applicable, "THREAT DETECTED".
3. **Verify AWS CloudWatch:**
    - Log in to the AWS Management Console and navigate to the CloudWatch service in the eu-west-1 (Ireland) region.
    - Go to **Dashboards** and select the RealTimeIDS dashboard. You should see widgets populating with data.
    - Go to **Alarms > All alarms**. You should see the IDS-* alarms in an 'OK' or 'Insufficient data' state.
4. **Verify AWS S3 Security Lake:**
    - Navigate to the S3 service in the AWS Console.
    - Open the albin-thesis-security-lake-2025 bucket.
    - Inside, you should see a folder structure being created (e.g., threat-detections/region=eu-west-1/...). As threat events are detected, new OCSF-formatted JSON files will appear here.

# 10 System Shutdown

To stop the system:

1. Go to the terminal window running flask_app.py and press Ctrl+C.
2. Go to the terminal window running real_time_ids.py and press Ctrl+C.

This will gracefully shut down both the web server and the IDS engine.

# 11 Conclusion

By following this configuration manual, you have successfully deployed a sophisticated, cloud-integrated Intrusion Detection System. The key components, including the high- accuracy machine learning model, the real-time analysis engine, the serverless inference pipeline, and the comprehensive monitoring and logging framework, are now fully operational.

The system is ready for live testing, demonstration, and further development. You can now observe its real-time detection capabilities via the web dashboard and monitor its performance and security posture through the integrated AWS services.