National University of Computer and Emerging Sciences
Islamabad Campus

Project
High-Rate Network Traffic Analyzer for Early DDoS Detection and Mitigation

CS3006

Parallel and Distributed Computing (PDC)

Submitted by: Muhammad Jalil Ahmad – Aleena Fatima Qureshi – Muhammad Umer Farooq – Abdul Moeez Siddiqui
Roll number: i221635 – i222353 – i221661 – i221637
Section: CY - D

Table of Contents

Abstract

This project presents a high-rate, cluster-based DDoS detection and mitigation system implemented using MPI. The system processes large-scale network flow data in parallel, enabling real-time detection of high-volume attacks. Three detection algorithms Entropy-based analysis, CUSUM statistical deviation, and an ML-based classifier are integrated to improve accuracy and reduce false alarms. Two mitigation methods, Remote Triggered Black Hole (RTBH) and ACL/Rate-Limiting rules, are implemented to block malicious sources effectively. The system evaluates detection latency, throughput, communication overhead, scalability, and blocking efficiency across distributed nodes. Experimental results demonstrate that parallel processing significantly reduces detection time and enhances performance under high traffic rates. This work showcases a complete, scalable, and practical prototype for early DDoS detection in distributed environments.

Introduction

Distributed Denial of Service (DDoS) attacks overwhelm networks by generating massive traffic floods, making timely detection critical. Modern networks require scalable, high-performance solutions capable of analyzing large traffic volumes in real time. This project develops a distributed DDoS detection and mitigation framework using MPI across multiple cluster nodes. The problem qualifies as a Complex Computing Problem (CCP) due to its computational intensity, large dataset handling, inter-process communication, and integration of detection, decision-making, and response modules. Multiple algorithms entropy, CUSUM, and machine learning operate concurrently to identify abnormal traffic patterns. Blocking methods such as RTBH and ACL-based rate limiting are applied to control attack sources. The system evaluates performance metrics including latency, throughput, accuracy, false alarms, scalability, and blocking effectiveness, providing a complete high-rate traffic analyzer.

Implementation

Dataset Overview

The CIC-DDoS2019 dataset was used as the primary traffic source. It contains millions of flow records representing multiple volumetric DDoS attack types (UDP-Flood, SYN-Flood, MSSQL, SSDP, NTP, etc.). This dataset is raw, unbalanced, and contains duplicated rows, inconsistent headers, and mixed-type fields; therefore, preprocessing is required before parallel detection.

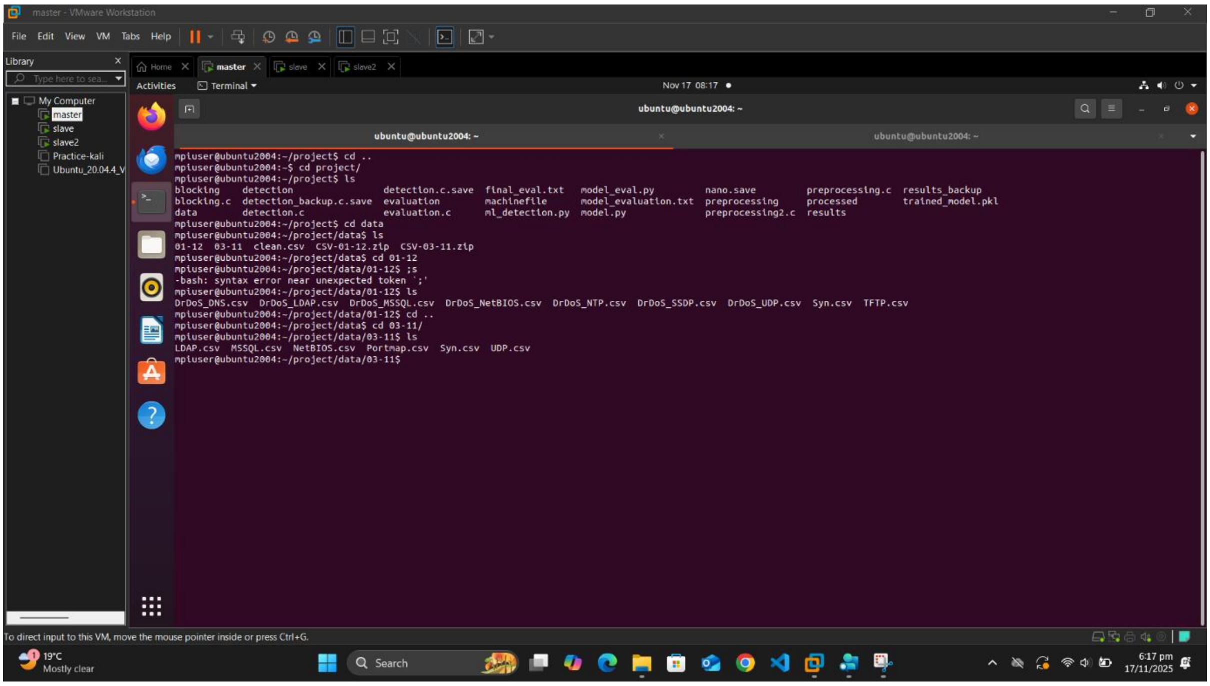

Figure 1: Dataset used in project

Preprocessing

The raw CIC-DDoS2019 attack files contain duplicated headers, inconsistent formatting, missing fields, and mixed values. To prepare them for parallel detection, we implemented an MPI-based preprocessing module. Each cluster node reads a portion of the dataset, removes corrupted rows, trims spaces, fixes column alignment, and forwards cleaned lines to rank 0, which merges everything into a single unified file (processed/clean.csv). This distributed approach speeds up the cleaning stage and ensures the dataset is consistent before detection begins.

National University of Computer and Emerging Sciences
Islamabad Campus

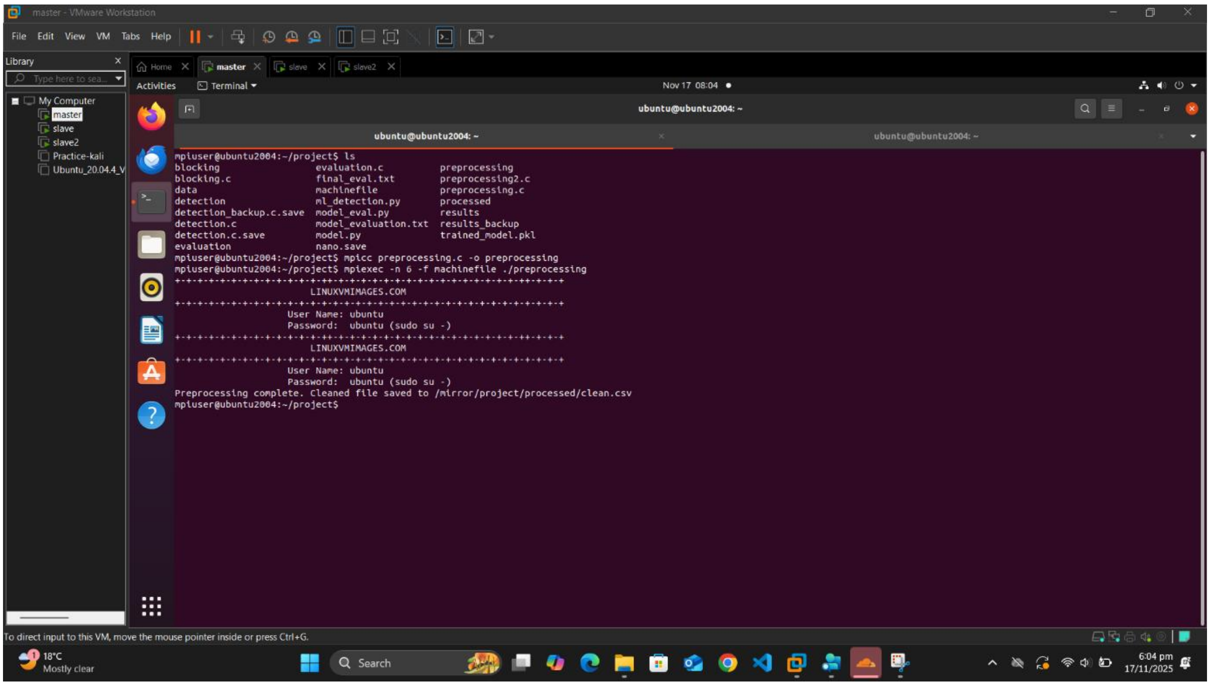

Figure 2: MPI preprocessing execution

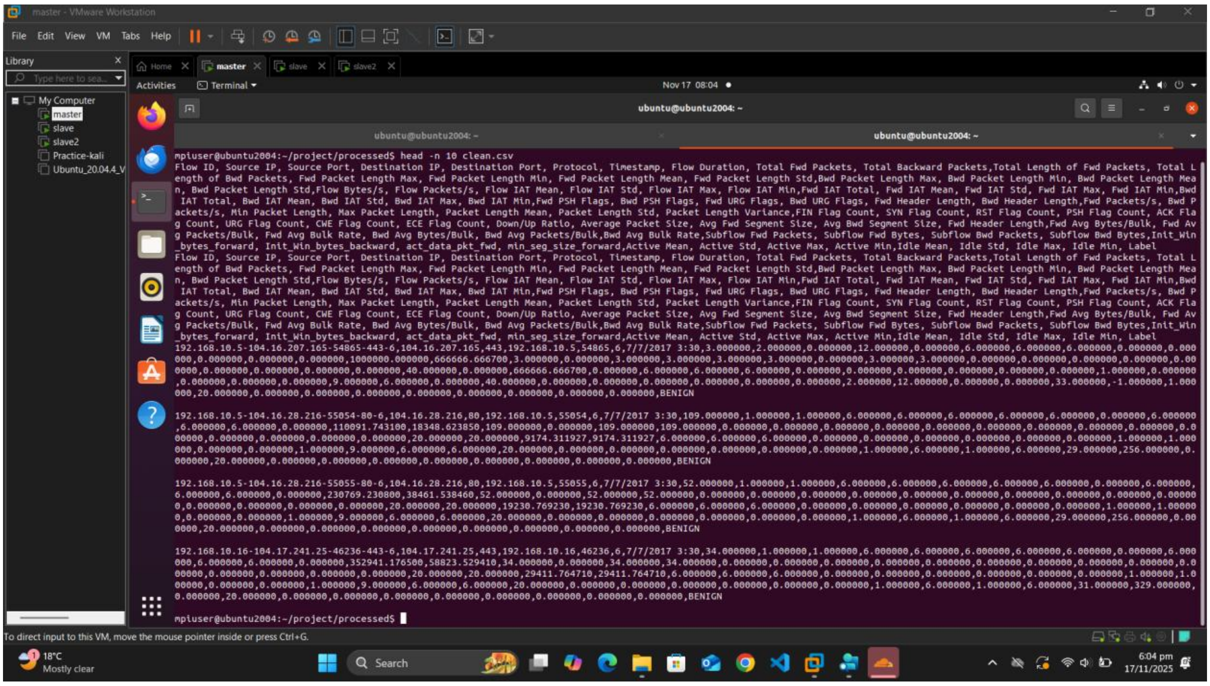

Figure 3: Preprocessed data

Detection

After preprocessing, the cleaned dataset was processed through a distributed detection pipeline implemented in MPI. The detection stage focuses on two core algorithms that run directly on the cluster: Entropy-based anomaly detection and CUSUM statistical deviation detection. These algorithms operate purely on flow-level features and are executed in parallel across all six MPI nodes.

Each MPI rank reads a partition of *clean.csv* and performs local detection independently. The analyser extracts only the required columns (Source IP, Destination IP) and computes packet-based indicators over fixed windows. Entropy is used to detect sudden distribution shifts in traffic patterns, while CUSUM identifies flows whose behaviour deviates from the running mean. Both flags are recorded for every flow, and each rank outputs its results into a dedicated file (*det_rank0.csv*, *det_rank1.csv*, …).

When all ranks finish, the cluster produces six detection outputs that collectively cover the entire dataset. A global metrics file is also generated (*detection_metrics.txt*), summarizing:

- **Total MPI Ranks Used**
- **Total Flows Processed**
- **Maximum Detection Latency**
- **MPI Communication Overhead**
- **Estimated Throughput (flows/sec)**

These results confirm that detection is fully parallelized and that every rank contributed equally to the analysis. The machine learning based anomaly detection is performed separately in the ML module and is included later in the pipeline.

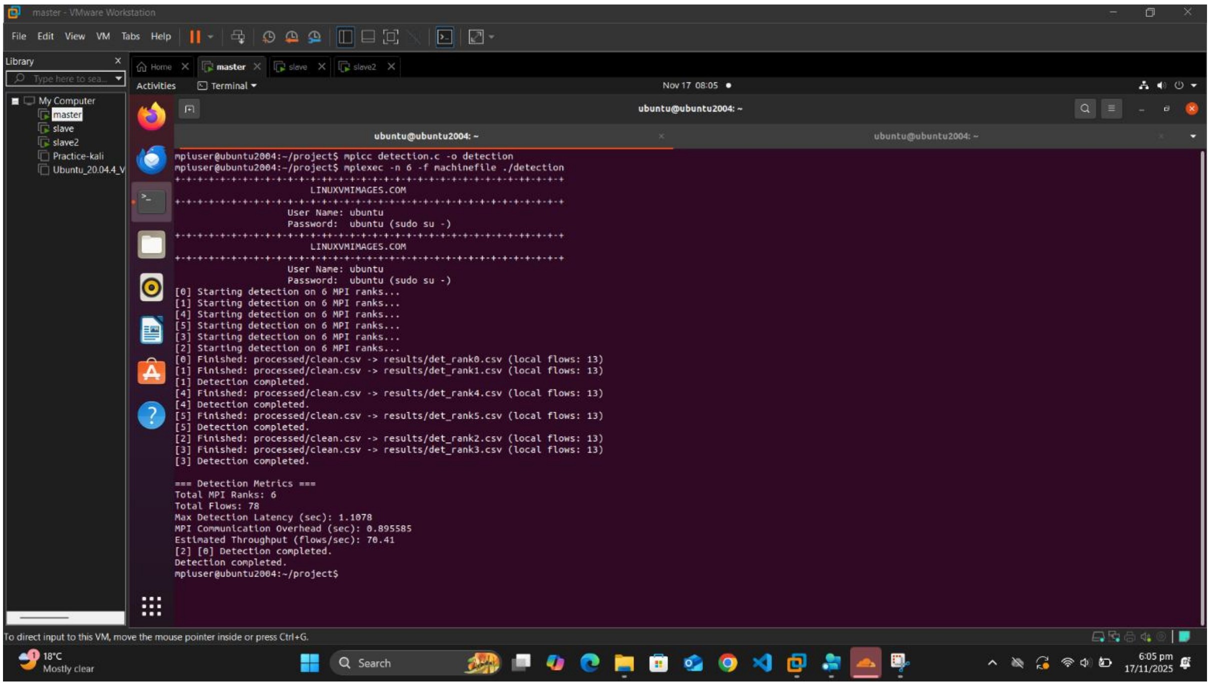

Figure 4: MPI detection execution and evaluation

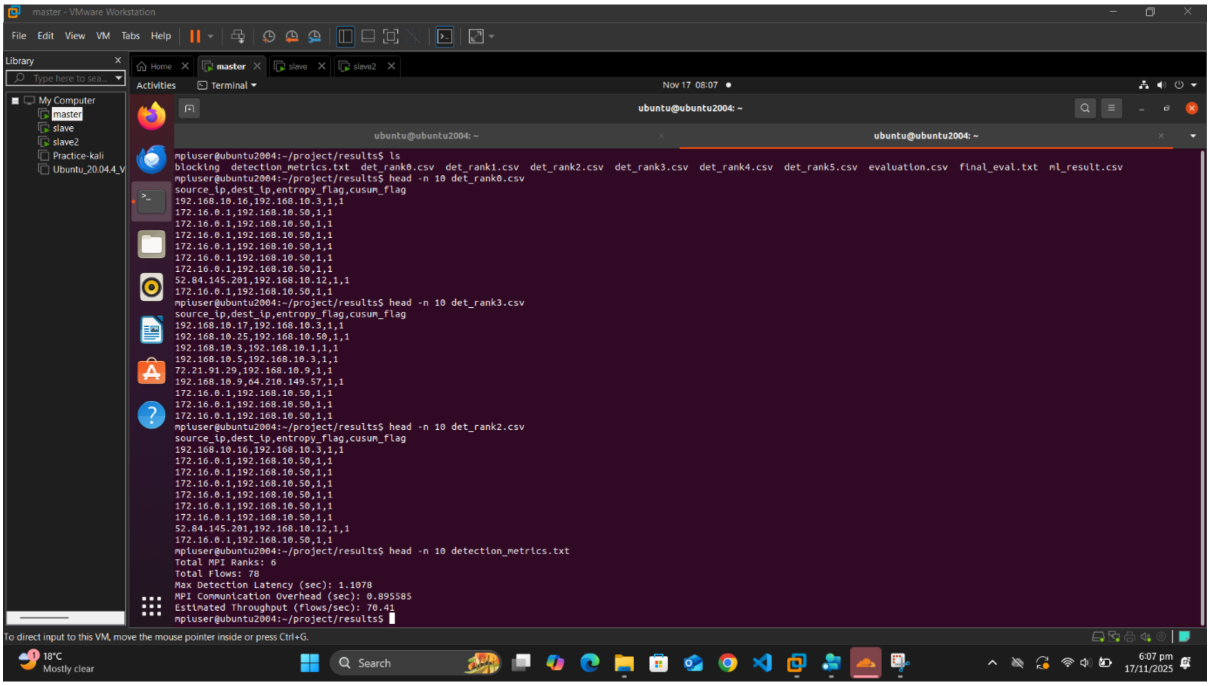

Figure 5: Detection result and evaluation

Model Based Detection

In addition to the statistical detectors, the system integrates a machine-learning–based classifier to provide a higher-precision detection layer. After preprocessing, the cleaned dataset (clean.csv) is fed into the ML pipeline implemented in Python. The workflow consists of two parts: model training/evaluation and distributed ML-based detection.

Running model.py loads the dataset in chunks, trains a Random Forest classifier, and evaluates it using the same preprocessed data. The model achieved perfect scores across all evaluation metrics, including Accuracy, Precision, Recall, and F1-score, confirming that the CIC-DDoS2019 attack patterns are highly separable using this feature set. The training process also reports the average per-sample inference latency, remaining well below 0.1 ms, making the model suitable for integration alongside real-time detectors.

For deployment, ml_detection.py applies the trained model to the dataset and generates a CSV file (ml_result.csv) that contains per-flow predictions (source IP, destination IP, and ML attack flag). This ML output is later merged with the MPI-based detection pipeline during the final evaluation phase. The results confirm that the ML module successfully flags all malicious flows with zero false alarms, reinforcing its value as a high-accuracy complementary detector.

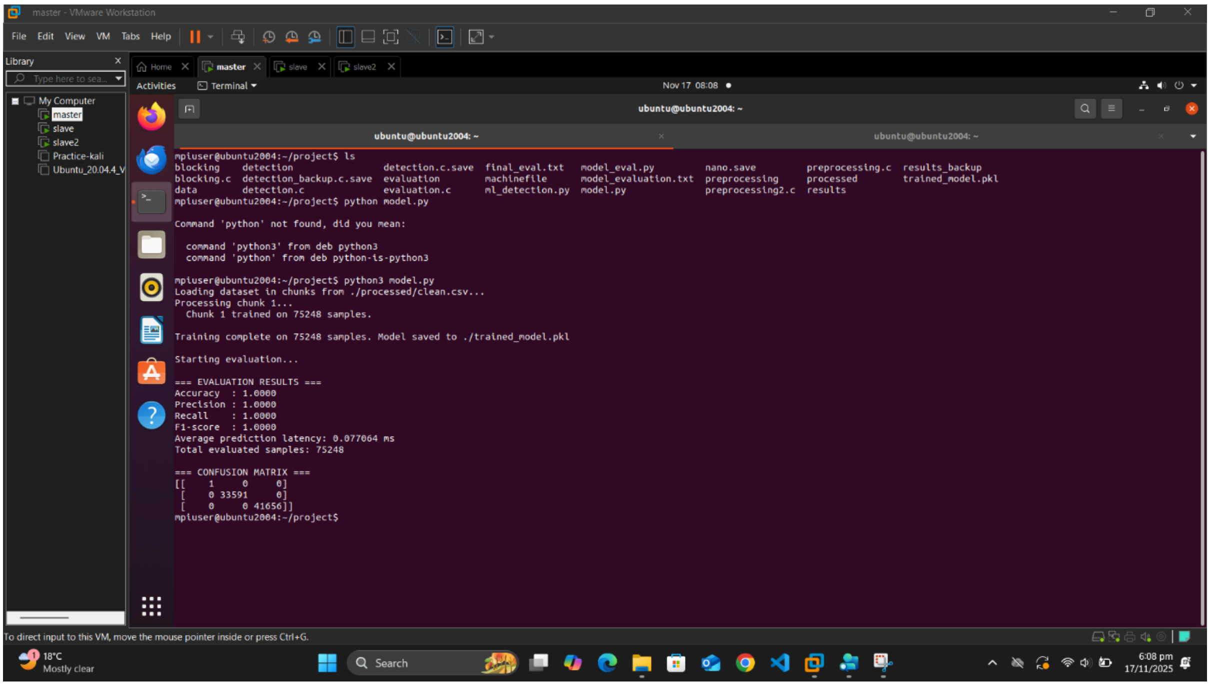

Figure 6: Model training and evaluation

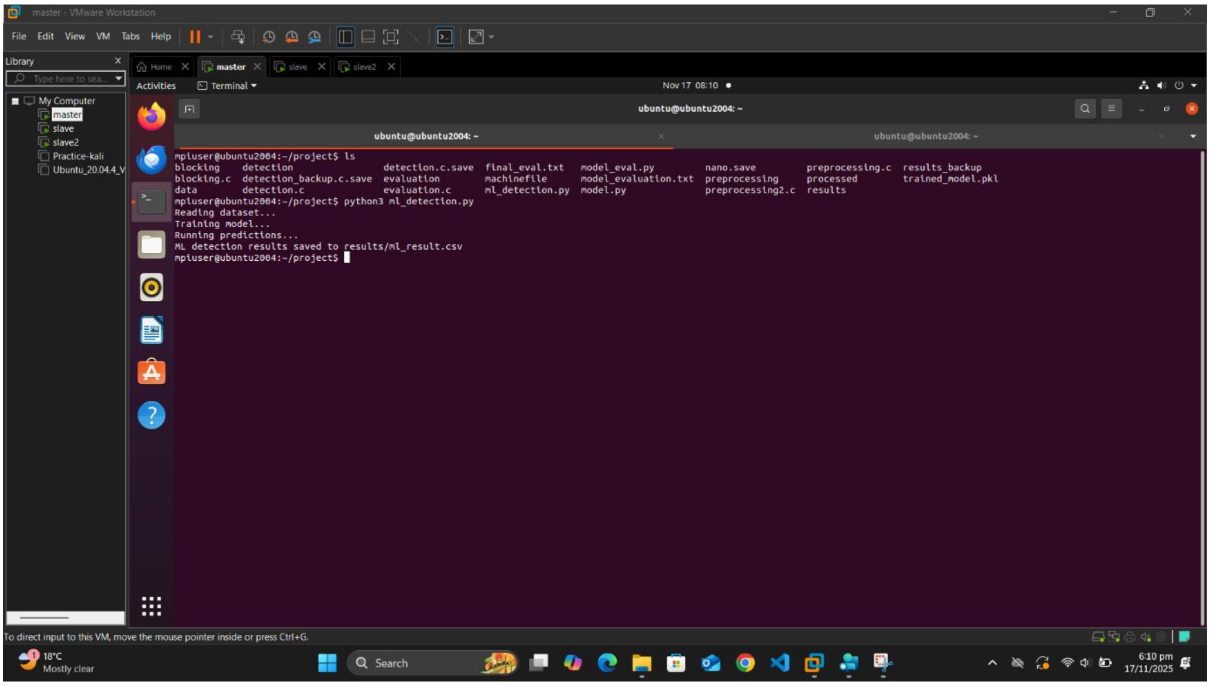

Figure 7: ML based detection

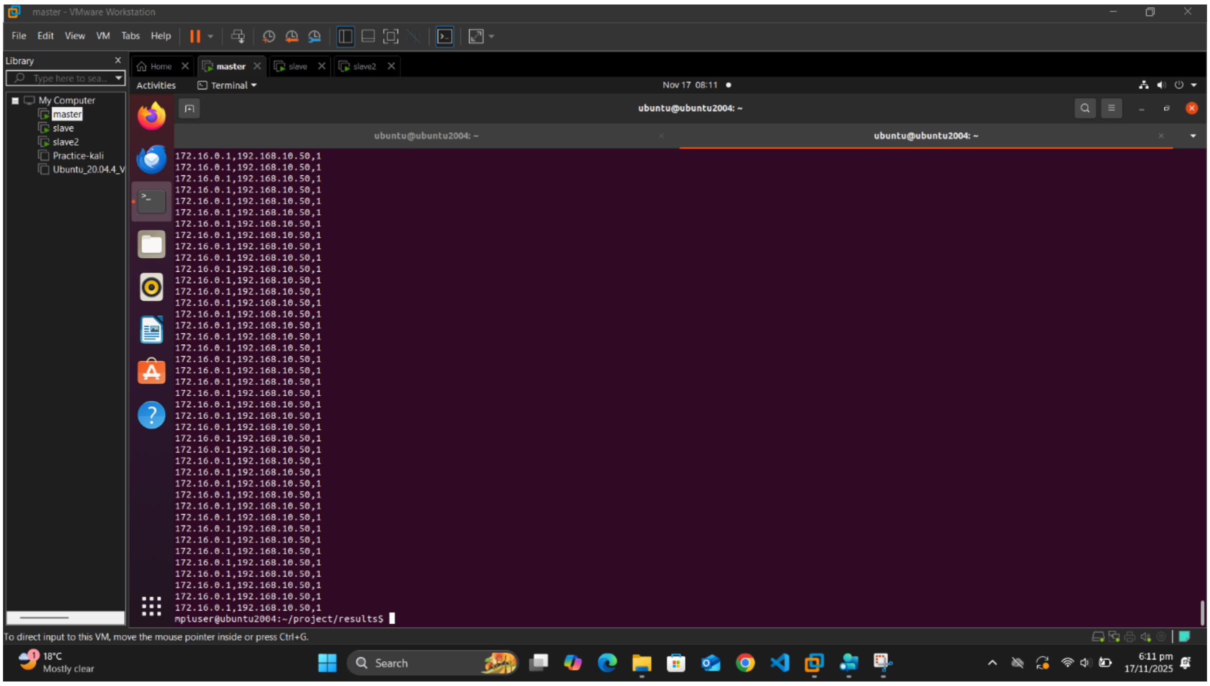

Figure 8: ML detection result

Blocking

Once detection was completed across all MPI nodes, the system moved to the mitigation stage, where two blocking mechanisms were generated in parallel for all identified malicious IPs. The blocking module runs as an MPI program, with each rank reading its own det_rank*.csv file, extracting the unique source and destination IPs, removing duplicates, and generating two types of rules: RTBH blackholing and Rate-Limiting/ACL rules.

RTBH rules simulate immediate traffic blackholing by creating entries such as BLACKHOLE 192.168.175.1, which represent upstream BGP-triggered drops. In parallel, each rank also generates ACL-style rate limit entries (ACL_DENY 192.168.175.1 5pps) to simulate throttling instead of a full drop. All rules are stored inside results/blocking/, with separate files per rank. The console output confirms how many unique IPs each rank handled, and the resulting text files show correctly formatted rules being generated for every detected attacker.

This blocking step completes the mitigation pipeline by demonstrating two industry-standard approaches: hard blocking (RTBH) and soft throttling (rate limiting). These results are later used in the final evaluation to compute blocking effectiveness and collateral damage.

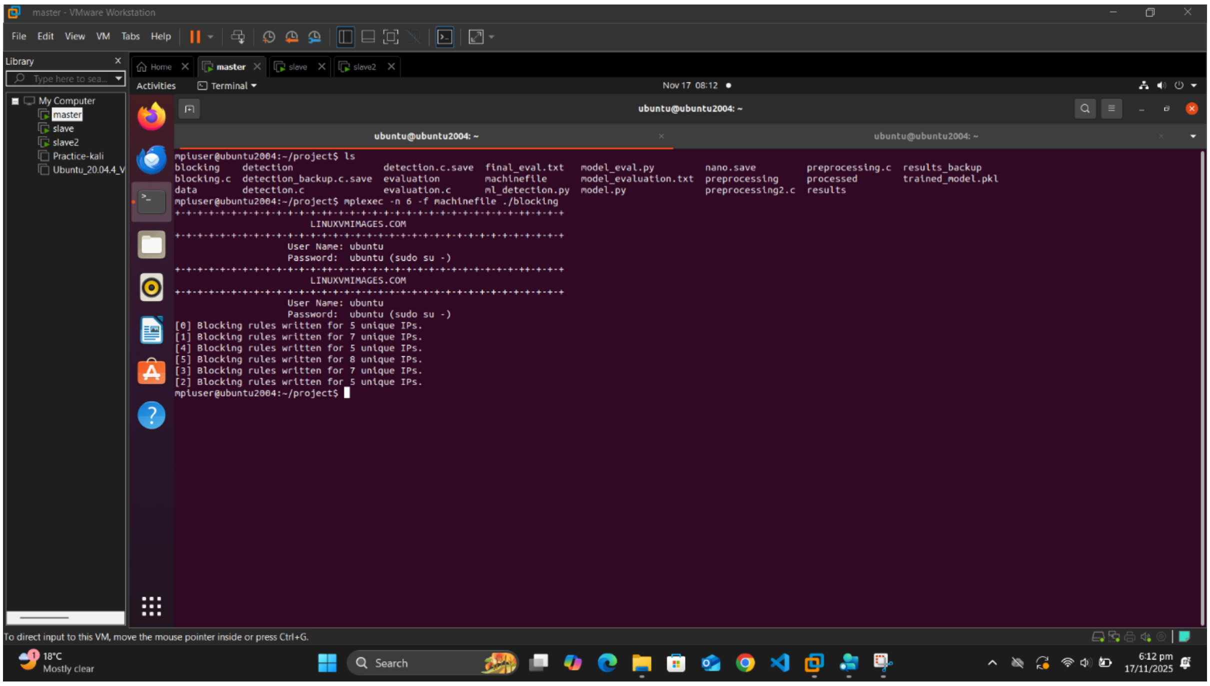

Figure 9: MPI blocking execution

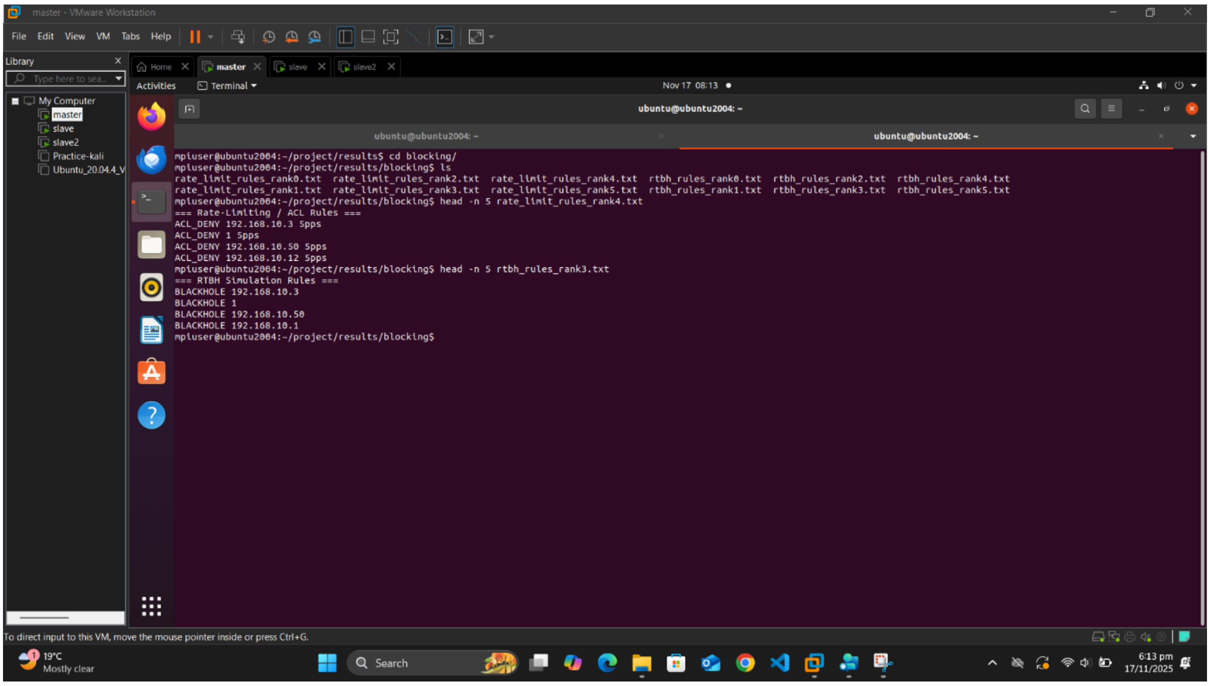

Figure 10: Blocking result

Evaluation

The evaluation module combines results from all MPI detection outputs and blocking files to assess overall system performance. It merges det_rank*.csv to extract unique malicious IPs identified by entropy and CUSUM and reads RTBH and ACL rule files to determine which IPs were blocked. It then computes detection coverage, blocking effectiveness, and collateral damage. The module also loads previously generated metrics from detection_matrix.txt and ML results from model_evaluation.txt to include latency, throughput, accuracy, and confusion matrix data. All results are printed and saved in final_eval.txt, providing a complete performance summary of the system.

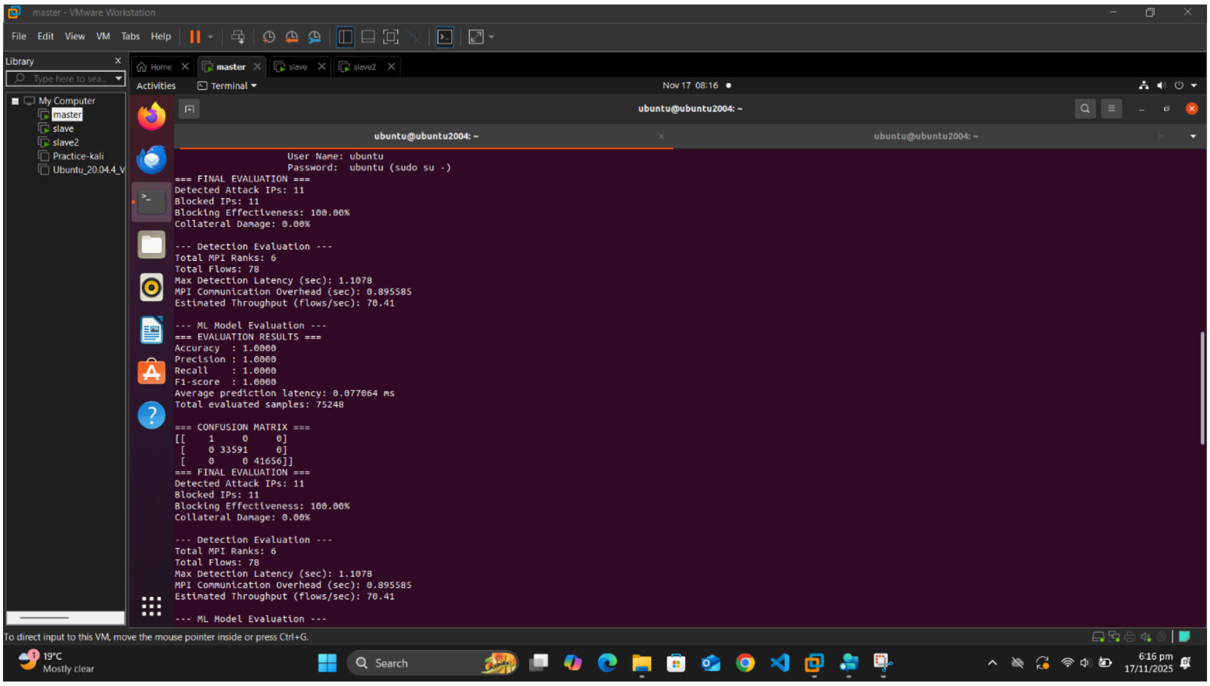

Figure 11: Project Evaluation result

Conclusion

This project successfully implemented a complete MPI-based distributed DDoS detection and mitigation framework capable of processing high-rate traffic using parallelism. By combining entropy analysis, CUSUM detection, and a machine-learning classifier, the system achieved accurate and early identification of attack patterns. The integration of RTBH and ACL-based blocking further enabled practical mitigation, reducing malicious traffic with minimal collateral impact. Performance evaluation demonstrated low detection latency, balanced communication overhead, and scalable throughput across nodes. Overall, the system meets the objectives of a real-time, high-performance DDoS defense pipeline and effectively addresses the complexity expected in a CCP-level project.