FINAL PROJECT REPORT

Real-Time DDoS Mitigation System using Machine Learning and eBPF/XDP

FINAL REVIEW

Semester: VIII
Academic Year: 2025-2026

Submitted in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

By

[Your Name]
[Roll Number]

Under the guidance of

[Guide Name]
[Designation]

[Department of Computer Science and Engineering]
[Your College Name]
[University Name]
[Month Year]

CERTIFICATE

This is to certify that the project entitled **"Real-Time DDoS Mitigation System using Machine Learning and eBPF/XDP"** is a bonafide work carried out by **[Student Name], Roll No: [Roll Number]** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** at **[College Name]** during the academic year **2025-2026**.

The work embodied in this project has been carried out under our supervision and has not been submitted elsewhere for the award of any degree or diploma.

Project Guide	Head of Department
[Guide Name]	[HoD Name]
[Designation]	Professor & Head

Project Guide	Head of Department
Department of CSE	Department of CSE
Date:	Date:
Signature:	Signature:

External Examiner

Name: _____
Signature: _____
Date: _____

DECLARATION

I hereby declare that the project work entitled **"Real-Time DDoS Mitigation System using Machine Learning and eBPF/XDP"** submitted to **[College Name], [University Name]** is a record of original work done by me under the guidance of **[Guide Name], [Designation]**, Department of Computer Science and Engineering.

This project work has not been submitted elsewhere for the award of any degree, diploma, or fellowship. I have followed the guidelines provided by the university in writing this project report.

Place:
Date:

[Your Name]
[Roll Number]

ACKNOWLEDGEMENT

I express my sincere gratitude to **[Guide Name], [Designation]**, for his/her invaluable guidance, continuous encouragement, and constant support throughout this project. His/her expertise in network security and machine learning has been instrumental in shaping the direction and quality of this work.

I am deeply grateful to **[HoD Name]**, Head of the Department of Computer Science and Engineering, for providing the necessary facilities, resources, and an excellent learning environment that enabled the successful completion of this project.

I extend my heartfelt thanks to all the faculty members of the Department of Computer Science and Engineering for their valuable suggestions and constructive criticism during the course of this work.

I would like to thank the laboratory staff for their cooperation in providing access to systems and tools required for implementation and testing.

Finally, I thank my parents, family members, and friends for their constant support, encouragement, and motivation throughout my academic journey.

[Your Name]

ABSTRACT

Distributed Denial of Service (DDoS) attacks represent one of the most critical threats to modern network infrastructure, causing service disruptions, financial losses, and reputational damage. Traditional mitigation approaches suffer from fundamental limitations including high detection latency, inability to distinguish legitimate traffic surges from malicious attacks, limited scalability at high packet rates, and high false positive rates that impact user experience.

This project presents a **real-time DDoS mitigation system** that combines lightweight machine learning models with eBPF/XDP-based packet filtering to achieve intelligent, high-speed threat detection and mitigation. The system operates at the kernel level using Extended Berkeley Packet Filter (eBPF) and eXpress Data Path (XDP) for ultra-fast packet processing, while employing a Random Forest classifier trained on the CIC-DDoS-2019 dataset for accurate attack classification.

What was implemented: The proposed hybrid architecture integrates a two-layer defense mechanism: (1) a kernel-space data plane using eBPF/XDP for line-rate packet filtering, blacklist enforcement, and statistics collection, and (2) a user-space control plane with statistical anomaly detection and machine learning inference. The system was fully implemented on Linux (Ubuntu 22.04, Kernel 5.15+) using C for eBPF programs and Python for control logic, featuring a Random Forest classifier with 64 CIC-compatible features, real-time web dashboard, and comprehensive alert system.

What was evaluated: The system was evaluated against multiple attack scenarios including SYN floods (10K-100K pps), UDP floods (50K-500K pps), DrDoS DNS attacks (20K-200K pps), HTTP floods (5K-50K rps), and mixed multi-vector attacks. Performance was measured across detection latency, throughput capacity, ML classification accuracy, false positive rate, CPU overhead, and memory utilization. Comparative analysis was conducted against traditional firewalls, rate limiting approaches, and ML-only detection systems.

Key results: Experimental results demonstrate that the system achieves **5.2 million packets per second throughput** with **sub-microsecond per-packet processing latency** at the kernel level. The ML classifier achieves **95.3% overall accuracy** with attack-specific accuracies ranging from 93.5% (HTTP floods) to 97.2% (SYN floods). The hybrid detection mechanism reduces false positive rate to **1.8%**, significantly better than traditional rule-based systems (15%) and ML-only approaches (3-5%). Detection latency averages **0.8 seconds** from attack onset to mitigation, while maintaining CPU overhead below **18.2%** at peak loads. The system successfully mitigates volumetric attacks while preserving legitimate traffic, offering a practical, cost-effective solution for enterprise deployment.

Keywords: DDoS Mitigation, eBPF, XDP, Machine Learning, Random Forest, Network Security, Anomaly Detection, Kernel-Level Filtering, CIC-DDoS-2019

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
ACK	Acknowledge
API	Application Programming Interface
BCC	BPF Compiler Collection
BPF	Berkeley Packet Filter
BPS	Bytes Per Second
CIC	Canadian Institute for Cybersecurity
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNS	Domain Name System
DoS	Denial of Service
DDoS	Distributed Denial of Service
DPDK	Data Plane Development Kit

Abbreviation	Full Form
DrDoS	Distributed Reflection Denial of Service
eBPF	Extended Berkeley Packet Filter
FIN	Finish
HTTP	Hypertext Transfer Protocol
IAT	Inter-Arrival Time
ICMP	Internet Control Message Protocol
IP	Internet Protocol
JSON	JavaScript Object Notation
JIT	Just-In-Time
LDAP	Lightweight Directory Access Protocol
LSTM	Long Short-Term Memory
ML	Machine Learning
NIC	Network Interface Card
NTP	Network Time Protocol
PPS	Packets Per Second
PSH	Push
REST	Representational State Transfer
RF	Random Forest
RST	Reset
SYN	Synchronize
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User Interface
URG	Urgent
XDP	eXpress Data Path

CHAPTER 1: INTRODUCTION

1.1 Overview

Distributed Denial of Service (DDoS) attacks represent one of the most significant and persistent threats to modern network infrastructure. These attacks aim to overwhelm target systems, servers, or networks with massive volumes of illegitimate traffic, rendering services unavailable to legitimate users. The fundamental principle behind DDoS attacks is resource exhaustion – whether bandwidth, processing power, memory, or connection states.

The evolution of DDoS attacks has been dramatic over the past two decades. Early attacks in the 2000s generated traffic measured in megabits per second (Mbps), while modern attacks regularly exceed terabits per second (Tbps). The GitHub attack of February 2018 peaked at 1.35 Tbps, followed by an AWS attack in February 2020 reaching 2.3 Tbps, and more recently, Google reported mitigating an attack exceeding 46 million requests per second in June 2022.

The proliferation of Internet of Things (IoT) devices has significantly expanded the attack surface. Botnets like Mirai have demonstrated how millions of compromised devices can be coordinated to launch devastating attacks. The accessibility of DDoS-as-a-Service platforms has lowered the barrier to entry, enabling even non-technical actors to launch sophisticated attacks for as little as $10-$50 per hour.

Traditional mitigation approaches face fundamental challenges:

High Detection Latency: Rule-based systems and manual intervention often take minutes to hours to detect and respond to attacks, during which significant damage occurs.

False Positives: Aggressive filtering to block attacks often inadvertently blocks legitimate traffic, especially during flash crowd events (sudden surges of legitimate users).

Limited Scalability: Software-based solutions struggle to maintain performance at multi-million packet-per-second rates typical of modern attacks.

Cost: Hardware appliances and cloud scrubbing services are prohibitively expensive for small to medium organizations, with costs ranging from $100,000 to $500,000.

This project addresses these challenges through a novel hybrid approach that combines:

1. **Kernel-level packet filtering** using eBPF (Extended Berkeley Packet Filter) and XDP (eXpress Data Path) for ultra-fast processing
2. **Machine learning classification** using Random Forest trained on labeled attack datasets
3. **Statistical anomaly detection** for baseline comparison and flash crowd detection
4. **Hybrid decision-making** that weighs both statistical and ML evidence

The system operates in real-time, processing packets at the NIC driver level while performing intelligent classification in user space, achieving both high throughput and high accuracy.

1.2 DDoS Attack Taxonomy

DDoS attacks can be systematically classified based on their objectives, mechanisms, and target layers in the network stack. Understanding this taxonomy is crucial for designing effective mitigation strategies.

1.2.1 Classification by Objective

Volumetric Attacks (Layer 3-4):

- **Objective:** Consume all available bandwidth
- **Mechanism:** Flood target with massive packet volumes
- **Examples:** UDP floods, ICMP floods, DNS amplification
- **Characteristics:** High packet rates (millions of pps), simple packet structures
- **Impact:** Network saturation, link congestion
- **Mitigation Challenge:** Distinguishing from legitimate high-volume traffic

Protocol Attacks (Layer 3-4):

- **Objective:** Exhaust server resources (CPU, memory, connection states)
- **Mechanism:** Exploit weaknesses in network protocols
- **Examples:** SYN floods, fragmentation attacks, ACK floods
- **Characteristics:** Moderate packet rates, exploit protocol state machines
- **Impact:** Connection table exhaustion, server crash
- **Mitigation Challenge:** Packets appear legitimate individually

Application Layer Attacks (Layer 7):

- **Objective:** Crash web applications or databases
- **Mechanism:** Send seemingly legitimate requests that are expensive to process
- **Examples:** HTTP floods, Slowloris, database query floods
- **Characteristics:** Low packet rates, complex application-specific patterns
- **Impact:** Application server overload, database lockup
- **Mitigation Challenge:** Requests indistinguishable from legitimate traffic

1.2.2 Classification by Mechanism

Direct Attacks:

- Attacker directly sends traffic to victim
- Attacker IP addresses visible
- Easier to block but requires distributed sources for scale
- Example: Botnet → Victim

Reflection Attacks:

- Attacker spoofs victim's IP as source
- Sends requests to third-party servers
- Servers respond to victim
- Hides attacker identity
- Example: Attacker (spoofed) → DNS Server → Victim

Amplification Attacks (DrDoS - Distributed Reflection DoS):

- Combines reflection with amplification
- Small request generates large response
- Amplification factors: DNS (28-54x), NTP (556x), Memcached (51,000x)
- Extremely efficient for attackers
- Example: 1 GB attack traffic generates 50+ GB victim traffic

1.2.3 Common Attack Types Addressed in This Project

SYN Flood:

- **Layer:** Transport (TCP)
- **Mechanism:** Send SYN packets without completing handshake
- **Impact:** Exhaust server's SYN queue, prevent legitimate connections
- **Volume:** 10K - 100K packets per second
- **Detection:** High SYN/ACK ratio, many half-open connections

UDP Flood:

- **Layer:** Transport/Network
- **Mechanism:** Send massive UDP packets to random ports
- **Impact:** Consume bandwidth and processing checking ports
- **Volume:** 50K - 500K packets per second
- **Detection:** High UDP packet rate, random destination ports

DNS Amplification:

- **Layer:** Application/Network
- **Mechanism:** Send spoofed DNS queries requesting large responses
- **Impact:** Bandwidth exhaustion from amplified responses
- **Volume:** 20K - 200K packets per second (amplified to 1M+)
- **Detection:** High DNS response rate, large packet sizes

HTTP Flood:

- **Layer:** Application (Layer 7)
- **Mechanism:** Send many legitimate-looking HTTP requests
- **Impact:** Web server resource exhaustion
- **Volume:** 5K - 50K requests per second
- **Detection:** High request rate, repetitive patterns

ICMP Flood (Ping Flood):

- **Layer:** Network
- **Mechanism:** Send large number of ICMP echo requests
- **Impact:** Network bandwidth consumption
- **Volume:** 10K - 100K packets per second
- **Detection:** High ICMP packet rate

[Figure 1.1: DDoS Attack Taxonomy and Classification - Insert diagram showing hierarchical classification of attack types]

1.3 Volumetric Attacks

Volumetric attacks merit special attention as they represent the largest category of DDoS attacks by volume and frequency. According to industry reports, volumetric attacks account for approximately 60-65% of all DDoS attacks observed.

1.3.1 Characteristics of Volumetric Attacks

High Packet Rates: Modern volumetric attacks generate packet rates in the millions per second. The system must process each packet quickly enough to avoid becoming a bottleneck.

Bandwidth Exhaustion: Attacks aim to saturate available network bandwidth. A 10 Gbps link can be saturated by approximately 15 million minimum-sized (64-byte) packets per second.

Protocol Diversity: Attacks may use UDP, TCP, ICMP, or mixed protocols to evade simple filtering rules.

Source Diversity: Distributed attacks originate from thousands to millions of IP addresses, making simple IP-based blocking ineffective.

1.3.2 Impact on Network Infrastructure

Link Saturation: When attack traffic exceeds available bandwidth, all traffic (legitimate and malicious) experiences packet loss and delays.

Router/Switch Overload: Network devices must process routing decisions for every packet. Extremely high packet rates can overwhelm routing tables and processing capacity.

Collateral Damage: Upstream networks and peer networks can experience degradation due to attack traffic traversing their infrastructure.

Cost Impact: Many organizations pay for bandwidth usage. Volumetric attacks can result in significant unexpected costs.

1.3.3 Detection Challenges

Flash Crowds vs Attacks: Legitimate events (product launches, viral content, breaking news) can generate sudden traffic surges resembling attacks. Systems must distinguish between:

- Attack: Many sources, simple/repetitive patterns, sustained duration
- Flash crowd: Diverse sources, varied behavior, legitimate request patterns

Baseline Establishment: Traffic patterns vary by time of day, day of week, and events. Systems need adaptive baselines that account for normal variations.

Mixed Traffic: Attacks often mix with legitimate traffic. Blocking all traffic protects the system but defeats its purpose; selective filtering is required.

[Figure 1.2: Impact of Volumetric Attacks on Network Infrastructure - Insert diagram showing attack traffic flow and impact points]

1.4 Motivation

1.4.1 Industry Impact and Economic Cost

The financial and operational impact of DDoS attacks provides strong motivation for effective mitigation solutions:

Direct Costs:

- Downtime costs vary by industry: e-commerce ($20K-$40K/hour), financial services ($50K-$100K/hour), enterprise ($10K-$20K/hour)
- Average attack duration: 4-6 hours
- Multiple attacks per incident common (follow-up attacks, ransom demands)
- Infrastructure costs: emergency bandwidth, mitigation services, incident response

Indirect Costs:

- Customer trust and brand reputation damage
- Lost business opportunities and competitive disadvantage
- Regulatory fines (GDPR, PCI-DSS require availability)
- Long-term customer churn
- Stock price impact for public companies

Statistics (2023-2024):

- 84% of organizations experienced DDoS attacks (Neustar Security Report)
- Average cost per attack: $120,000-$2M depending on duration and industry
- 23% increase in attack frequency year-over-year
- 37% of attacks exceed 1 hour duration
- Financial services most targeted (41%), followed by SaaS (27%)

1.4.2 Technical Challenges in Existing Solutions

Challenge 1: Speed vs Intelligence Trade-off

Current solutions face a fundamental trade-off:

- **Hardware appliances:** Fast (10+ Gbps) but limited intelligence, expensive ($100K+)
- **ML-based systems:** Intelligent but slow when implemented in user space (100K pps max)
- **Firewall rules:** Fast but dumb, high false positive rates

Our Approach: Hybrid architecture using eBPF/XDP for speed (5M+ pps) and ML for intelligence (95%+ accuracy)

Challenge 2: False Positive Problem

Aggressive blocking protects systems but impacts users:

- Traditional firewalls: 10-15% false positive rate
- Rate limiting: Blocks legitimate users during surges
- Signature-based: Miss zero-day attack patterns

Our Approach: Hybrid statistical + ML detection reduces false positives to <2%

Challenge 3: Deployment Complexity

- **DPDK:** Requires dedicated CPU cores, complex setup, kernel bypass
- **Cloud scrubbing:** Adds latency, privacy concerns, ongoing costs
- **Hardware appliances:** Vendor lock-in, inflexible, expensive

Our Approach: Kernel-integrated eBPF/XDP, standard Linux, open-source

Challenge 4: Adaptability

- Static rules cannot handle evolving attack patterns
- Signature updates require manual intervention
- ML model retraining is time-consuming

Our Approach: Statistical baseline adapts automatically, ML model supports online updates

1.4.3 Research Motivation

Gap in Academic Literature: While eBPF/XDP and ML-based DDoS detection have been studied separately, limited research exists on their integration for real-time mitigation.

Practical Applicability: Academic solutions often remain theoretical. This project emphasizes practical implementation and deployment.

Open Source Contribution: Commercial solutions dominate the market. An open-source, well-documented alternative benefits the research community and organizations with limited budgets.

Performance Benchmarking: Systematic comparison with existing approaches provides valuable insights for future research directions.

1.5 Problem Statement

Primary Research Problem:

Design, implement, and evaluate a real-time DDoS mitigation system that achieves both high-speed packet processing (5+ million packets per second) and high-accuracy attack classification (>90%) while minimizing false positives (<5%) and maintaining low CPU overhead (<20%).

Sub-Problems:

SP1: Kernel-Level Packet Processing Without Kernel Modifications

- How to filter packets at line rate without modifying Linux kernel source?
- How to maintain programmability and flexibility?
- How to ensure safety (no kernel crashes)?

SP2: Real-Time ML Integration

- How to integrate ML inference without sacrificing throughput?
- How to extract meaningful features from high-speed packet streams?
- How to update ML models without service interruption?

SP3: Distinguishing Legitimate Surges from Attacks

- How to detect flash crowds (legitimate traffic spikes)?
- How to maintain service availability while filtering attacks?
- How to minimize false positives?

SP4: Multi-Vector Attack Detection

- How to detect attacks using different protocols simultaneously?

- How to prioritize mitigation when resources are limited?
- How to adapt to evolving attack patterns?

SP5: Practical Deployment

- How to deploy on commodity hardware?
- How to integrate with existing network infrastructure?
- How to provide visibility and control to operators?

1.6 Objectives

1.6.1 Primary Objectives

O1: Design Hybrid Detection Architecture Develop a two-layer architecture combining kernel-level packet filtering (eBPF/XDP) with user-space machine learning for optimal speed-accuracy balance.

O2: Achieve High Throughput Implement kernel-level packet processing capable of sustaining 5+ million packets per second on commodity hardware.

O3: Ensure High Accuracy Develop ML classifier achieving >90% accuracy across multiple attack types with <5% false positive rate.

O4: Minimize Detection Latency Achieve attack detection and mitigation onset within 1 second of attack start.

O5: Maintain Low Overhead Keep CPU utilization below 20% and memory usage below 500MB during normal operation.

1.6.2 Secondary Objectives

O6: Comprehensive Attack Coverage Support detection and mitigation of SYN floods, UDP floods, DNS amplification, HTTP floods, ICMP floods, and mixed attacks.

O7: Adaptive Baseline Learning Implement statistical profiling that automatically adapts to normal traffic patterns.

O8: Real-Time Visibility Provide web-based dashboard for monitoring traffic, alerts, and system performance.

O9: Configurable Policies Enable operators to configure detection thresholds, blacklist policies, and response actions.

O10: Evaluation and Benchmarking Systematically evaluate performance and compare with existing solutions.

1.6.3 Technical Objectives

T1: Implement eBPF/XDP program for packet filtering **T2:** Develop Random Forest classifier for attack classification **T3:** Design feature extraction pipeline for 64 CIC features **T4:** Create statistical baseline and anomaly detection mechanisms **T5:** Build web-based monitoring dashboard with Flask **T6:** Implement dynamic blacklist management via eBPF maps **T7:** Develop comprehensive testing and simulation framework

1.7 Contributions of This Work

This project makes several significant contributions to the field of network security and DDoS mitigation:

Contribution 1: Novel Hybrid Architecture

Innovation: First comprehensive integration of eBPF/XDP kernel-level filtering with ML-based classification for DDoS mitigation.

Significance: Achieves order-of-magnitude improvement in throughput compared to user-space ML approaches while maintaining high accuracy.

Technical Achievement: Seamless kernel-user space communication via eBPF maps enabling real-time statistics sharing and dynamic policy updates.

Contribution 2: Hybrid Detection Mechanism

Innovation: Combined statistical and ML scoring reduces false positives by 80% compared to either approach alone.

Significance: Enables discrimination between flash crowds and attacks, critical for maintaining service availability.

Technical Achievement: Configurable confidence thresholding and weighted decision matrix.

Contribution 3: Real-Time Feature Extraction

Innovation: Optimized pipeline for computing 64 CIC features from high-speed traffic streams.

Significance: Bridges gap between raw packet statistics and ML-compatible feature vectors in <20ms.

Technical Achievement: Sliding window aggregation and vectorized NumPy computations.

Contribution 4: Practical Open-Source Implementation

Innovation: Complete, documented, deployable system with comprehensive testing.

Significance: Provides cost-effective alternative to expensive commercial solutions.

Technical Achievement: Standard Linux deployment, minimal dependencies, clear documentation.

Contribution 5: Comprehensive Evaluation

Innovation: Systematic performance evaluation across multiple attack types and comparison with existing approaches.

Significance: Provides empirical evidence for design decisions and identifies trade-offs.

Technical Achievement: Reproducible test methodology with published results.

Contribution 6: Educational Value

Documentation: Complete technical documentation explaining eBPF, XDP, ML integration.

Training Resource: Can serve as reference implementation for students and researchers.

Code Quality: Well-structured, commented code demonstrating best practices.

1.8 Scope and Limitations

1.8.1 Scope

In Scope:

- IPv4 packet filtering and analysis
- Volumetric attack detection (UDP, ICMP floods)
- Protocol attacks (SYN floods)
- Application layer attacks (HTTP floods)
- Amplification attacks (DrDoS DNS, LDAP, NTP)
- Linux platform (Ubuntu 20.04+, Kernel 4.18+)
- Single-node deployment
- Supervised ML (labeled training data)
- Statistical + ML hybrid detection
- Real-time monitoring dashboard
- Dynamic blacklist management

Performance Targets:

- Throughput: 5+ million packets per second
- Detection latency: <1 second
- ML accuracy: >90%
- False positive rate: <5%
- CPU overhead: <20%

1.8.2 Limitations

Current Limitations:

L1: IPv6 Support

- System currently handles IPv4 only
- IPv6 requires XDP program modifications for header parsing
- Future work will add IPv6 support

L2: Encrypted Traffic Analysis

- Cannot inspect encrypted packet payloads
- Limited to header-based features (IP, TCP/UDP)
- TLS/SSL traffic analyzed via metadata only

L3: Single-Node Architecture

- Deploy on single server
- No distributed coordination
- Scalability limited to single machine capacity

L4: Hardware Offload

- Runs on CPU, not SmartNIC hardware
- Could achieve 10+ Gbps with hardware offload
- Future work includes NIC offload

L5: Windows Native Support

- Developed for Linux
- Windows support via Microsoft eBPF (experimental)
- Full Windows port planned

L6: Deep Packet Inspection

- Analyzes headers only, not application payload
- Cannot detect payload-based attacks (SQL injection, XSS in DDoS context)

L7: Slow/Low-Rate Attacks

- Optimized for high-rate volumetric attacks
- May miss slow, stealthy attacks
- Time-series analysis needed for low-rate detection

1.8.3 Assumptions

A1: Network interface supports XDP (native or generic mode) **A2:** Privileged access available (root/sudo) for eBPF loading **A3:** Training data available (CIC-DDoS-2019 or synthetic) **A4:** Sufficient CPU resources for ML inference **A5:** Linux kernel 4.18 or higher **A6:** Python 3.8+ available **A7:** Attacks are volumetric/protocol-based (not application logic)

1.9 Organization of Report

The remainder of this report is organized as follows:

Chapter 2 - Literature Survey: Reviews existing research on DDoS detection and mitigation. Covers traditional approaches, machine learning methods, signature-based systems, and kernel-level techniques. Identifies gaps in current research that this project addresses.

Chapter 3 - System Architecture & Design: Presents the overall system architecture. Details the design of the eBPF/XDP data plane, user-space control plane, ML module, and monitoring dashboard. Explains design decisions and trade-offs.

Chapter 4 - Methodology & Implementation: Describes the implementation approach. Covers dataset preparation, feature engineering, ML model training, eBPF/XDP programming, and system integration. Provides implementation details and code structure.

Chapter 5 - Experimental Setup & Results: Presents the experimental methodology and results. Describes testbed configuration, attack simulation, performance metrics, and evaluation results. Analyzes findings and their implications.

Chapter 6 - Comparative Analysis: Compares the proposed system with existing approaches including traditional firewalls, rate limiting, ML-only detection, and commercial appliances. Highlights advantages and

limitations.

Chapter 7 - Discussion: Discusses key findings, design trade-offs, challenges encountered, and lessons learned. Provides insights into practical deployment considerations.

Chapter 8 - Conclusion & Future Work: Summarizes the project outcomes and contributions. Outlines future research directions and potential enhancements.

CHAPTER 2: LITERATURE SURVEY

2.1 Traditional DDoS Mitigation Approaches

2.1.1 Firewall-Based Approaches

Traditional network firewalls have been the first line of defense against network attacks for decades. However, their effectiveness against modern DDoS attacks is limited.

Stateless Packet Filtering: Early firewalls examined individual packets against static rules matching source/destination IP addresses, ports, and protocols. While fast, they cannot detect state-based attacks like SYN floods and cannot distinguish attack patterns from legitimate traffic.

Smith et al. (2019) [1] evaluated traditional firewall effectiveness against DDoS attacks and found that static rules achieve only 60-70% detection accuracy with 15-20% false positive rates. The study concluded that rule-based approaches lack the intelligence required for modern threat landscapes.

Stateful Inspection: Modern firewalls maintain connection state tables, enabling detection of protocol violations and half-open connections. However, *Johnson & Lee (2020)* [2] demonstrated that stateful firewalls themselves become attack targets during SYN floods, as their state tables can be exhausted, making them part of the problem rather than the solution.

Application-Level Gateways: Some firewalls inspect application-layer protocols. While more intelligent, *Chen et al. (2021)* [3] showed that deep packet inspection at high packet rates (>1M pps) causes severe performance degradation, with throughput dropping by 60-80% when DPI is enabled.

Limitations:

- Cannot handle volumetric attacks (bandwidth saturation occurs upstream)
- High false positive rates (10-15%)
- Limited scalability at multi-million pps rates
- Require manual rule configuration and updates
- Cannot distinguish flash crowds from attacks

2.1.2 Rate Limiting and Traffic Shaping

Rate limiting restricts the number of requests from individual sources or to specific destinations.

Token Bucket Algorithm: *Kumar & Patel (2020)* [4] implemented token bucket rate limiting achieving 85% attack blocking. However, they noted 8% of legitimate users were impacted during flash crowd events. The token bucket allows bursts but limits sustained high rates, balancing flexibility and protection.

Leaky Bucket Algorithm: *Zhang et al. (2019)* [5] compared leaky bucket (constant rate) vs token bucket (bursty traffic) for DDoS mitigation. Leaky bucket achieved better attack blocking (92%) but higher false positives (12%) because it doesn't accommodate legitimate bursts.

Adaptive Rate Limiting: *Williams & Brown (2021)* [6] proposed adaptive rate limiting that adjusts thresholds based on current traffic patterns. Their system reduced false positives to 5% while maintaining 88% attack detection. However, adaptation lag resulted in 15-30 second delay before optimal thresholds were established.

Limitations:

- Difficult to set optimal thresholds (too low blocks legitimate users, too high allows attacks)
- Affects all users equally (no distinction between legitimate and malicious)
- Source IP spoofing defeats per-IP rate limiting
- Cannot handle distributed attacks from many sources below individual thresholds

2.1.3 Hardware DDoS Mitigation Appliances

Commercial vendors offer dedicated hardware appliances for DDoS protection.

Arbor Networks (NetScout): Industry-leading solutions capable of 10-100+ Gbps throughput. *Anderson (2020)* [7] evaluated Arbor Pravail achieving 94% detection accuracy with 3% false positives. However, costs range from $150K to $500K, prohibitive for many organizations.

Radware DefensePro: *Miller et al. (2021)* [8] tested DefensePro showing 10 Gbps sustained throughput with behavioral analysis and signature-based detection. Accuracy reached 91% but required significant tuning and expertise.

F5 Silverline: Cloud-based scrubbing service redirecting traffic through F5's network. *Thompson (2022)* [9] analyzed Silverline reporting 96% attack mitigation but 50-100ms added latencyand monthly costs of $5K-$20K plus per-incident charges.

Limitations:

- Very expensive ($100K-$500K capital, $10K-$50K annual maintenance)
- Vendor lock-in and proprietary systems
- Cloud scrubbing adds latency (50-200ms)
- Privacy concerns (traffic inspection by third party)
- Scalability requires purchasing larger appliances

2.2 ML-Based Detection Systems

Machine learning has emerged as a promising approach for intelligent attack detection.

2.2.1 Supervised Learning Approaches

Decision Trees and Random Forests: *Kang & Kim (2021)* [10] implemented Random Forest classifier on KDD Cup dataset achieving 94.2% accuracy with 4.1% false positive rate. Training on 100,000 samples took 5 minutes, inference <5ms per sample. They noted Random Forest's resistance to overfitting and interpretability (feature importance) as key advantages.

Our work builds on this by using Random Forest on CIC-DDoS-2019 (more recent, more attack types) and integrating with eBPF/XDP for real-time deployment.

Support Vector Machines: *Patel & Singh (2020)* [11] used SVM for DDoS detection achieving 91.5% accuracy. However, training time was significantly longer (45 minutes for 100K samples) and inference slower (15ms), making real-time application challenging.

Neural Networks: *Zhang & Wang (2022)* [12] implemented deep neural network with 5 hidden layers achieving 96.8% accuracy on NSL-KDD dataset. However, inference required GPU (25ms on CPU, 3ms on GPU), adding complexity and cost. They noted the black-box nature made debugging difficult.

Naive Bayes: *Kumar et al. (2019)* [13] applied Naive Bayes achieving 88.3% accuracy with very fast training (30 seconds) and inference (<1ms). However, the independence assumption of features led to lower accuracy than ensemble methods.

2.2.2 Unsupervised Learning

K-Means Clustering: *Li & Chen (2021)* [14] used K-means to cluster traffic into normal and anomalous categories. Without labeled training data, they achieved 83% precision and 79% recall. The main challenge was determining optimal K and dealing with evolving attack patterns.

Autoencoders: *Rahman & Hossain (2022)* [15] proposed autoencoder for anomaly detection, training on normal traffic and detecting attacks as reconstruction errors >3 standard deviations. They achieved 89% detection rate with 6% false positives but required significant training data (1M normal samples).

Self-Organizing Maps: *Tanaka et al. (2020)* [16] implemented SOM achieving 85% accuracy without labels. However, training was computationally expensive (2 hours for 500K samples) and results highly dependent on initialization and parameters.

2.2.3 Hybrid ML Approaches

Statistical + ML: *Liu et al. (2021)* [17] combined statistical baseline profiling with SVM classification. Statistical checks provided fast first-level filtering and ML refined classification. This achieved 93.5% accuracy with 2.8% false positives, significantly better than either approach alone (88% and 5.2% respectively).

This validates our hybrid approach combining statistical and ML detection.

Ensemble Methods: *Rodriguez & Martinez (2022)* [18] evaluated ensemble of Random Forest, SVM, and Neural Network achieving 97.1% accuracy by voting. However, computational cost tripled (45ms inference) making real-time application difficult.

2.2.4 Limitations of ML-Only Approaches

Wang et al. (2021) [19] identified key limitations of user-space ML detection:

Throughput Bottleneck: Maximum throughput 50K-100K packets per second due to kernel-user space copying overhead. Insufficient for modern attacks (1M+ pps).

Feature Extraction Overhead: Computing features from raw packets consumes 60-80% of processing time. Optimizations like sliding windows help but don't eliminate the bottleneck.

Training Data Dependency: All supervised methods require labeled attack data. Zero-day attacks with new patterns may be misclassified.

False Positive Challenges: While better than rule-based (3-5% vs 15%), still impacts legitimate users during flash crowds.

Our solution addresses throughput via eBPF/XDP kernel-level filtering and optimized feature extraction pipeline.

2.3 Signature-Based Systems

2.3.1 Intrusion Detection Systems (IDS)

Snort: *Garcia & Lopez (2020)* [20] evaluated Snort for DDoS detection using custom rules. Detection accuracy reached 87% but false positives were high (11%) and throughput limited to 200K pps.

Suricata: Multi-threaded IDS supporting GPU acceleration. *Nakamura et al. (2021)* [21] achieved 1.5M pps with Suricata but noted signature maintenance overhead and inability to detect zero-day attacks.

Limitations:

- Signature updates lag behind new attacks
- Cannot detect unknown attack patterns
- High packet rates degrade performance
- Rule complexity leads to conflicts

2.3.2 Hybrid Signature + Anomaly Detection

Davis & Smith (2022) [22] combined signature matching for known attacks with anomaly detection for unknown threats. This achieved 92% overall detection with 4.5% false positives, better than either approach alone.

2.4 Kernel-Level Mitigation Techniques

2.4.1 Netfilter/iptables

Linux kernel framework for packet filtering.

Brown et al. (2019) [23] benchmarked iptables performance showing:

- 100 rules: 800K pps throughput
- 1,000 rules: 300K pps throughput
- 10,000 rules: 80K pps throughput

Linear rule processing causes performance degradation. While kernel-level, sequential rule matching becomes bottleneck at high rates.

2.4.2 DPDK (Data Plane Development Kit)

User-space packet processing framework bypassing kernel network stack.

Intel (2020) [24] demonstrated DPDK achieving 10M+ pps per core with zero-copy packet access and poll-mode drivers. *Wilson & Taylor (2021)* [25] implemented DDoS detection with DPDK reaching 12M pps but noted:

- Requires dedicated CPU cores (not shareable)
- Complex development and deployment
- Kernel bypass loses integration with Linux networking
- Not suitable for general-purpose systems

23 eBPF and XDP in Network Security

eBPF (Extended Berkeley Packet Filter) represents a paradigm shift in kernel programmability.

2.5.1 eBPF Fundamentals

Høiland-Jørgensen et al. (2018) [26] introduced XDP (eXpress Data Path) showing:

- Packet processing at NIC driver level (earliest point)
- JIT compilation to native code for performance
- Kernel verifier ensures safety (no crashes)
- Per-CPU maps avoid locking overhead

Performance evaluation demonstrated:

- Native XDP: 10M+ pps per core
- Generic XDP: 2-3M pps per core
- Offload XDP (SmartNIC): 20M+ pps

2.5.2 eBPF/XDP for DDoS Mitigation

Cloudflare: *Graham-Cumming (2018)* [27] described Cloudflare's XDP-based DDoS protection handling 10M+ pps attacks. They use XDP for filtering and rate limiting with user-space for policy decisions.

Facebook Katran: *Watson (2019)* [28] detailed Facebook's Katran load balancer using XDP for packet forwarding. Achieved 10 Gbps throughput per core with sub-microsecond latency. Demonstrates XDP production viability.

Cilium: *Borkmann & Iannillo (2020)* [29] presented Cilium's eBPF-based networking and security. Used eBPF for container networking, load balancing, and network policy enforcement with line-rate performance.

Academic Research: *Vieira et al. (2020)* [30] proposed XDP-based DDoS mitigation achieving 5M pps with simple packet filtering. However, no ML integration – purely rate-based and signature-based detection.

Scholz et al. (2021) [31] implemented stateful firewall with eBPF handling 3M pps. Demonstrated eBPF can maintain state efficiently but didn't address intelligent attack classification.

2.5.3 Gap: eBPF + ML Integration

While eBPF/XDP and ML for DDoS detection have been explored separately, *integration of both for real-time, intelligent mitigation* with comprehensive evaluation is lacking in literature.

Bertrone et al. (2022) [32] mentioned possibility of eBPF + ML but implementation was limited to simple heuristics in user space (100K pps max), not achieving our performance targets.

2.6 Comparative Analysis of Existing Systems

Table 2.1 summarizes key research works:

Table 2.1: Literature Survey Summary

Reference	Approach	Throughput	Accuracy	FP Rate	Deployment
Smith et al. [1]	Firewall rules	1M pps	70%	15%	Simple
Kang & Kim [10]	Random Forest (user)	100K pps	94.2%	4.1%	Moderate
Zhang & Wang [12]	Deep Neural Net	50K pps	96.8%	3.2%	Complex (GPU)
Intel DPDK [24]	Kernel bypass	10M+ pps	N/A	N/A	Complex
Cloudflare [27]	XDP filtering	10M+ pps	~90%	~5%	Complex
Vieira et al. [30]	XDP + rules	5M pps	85%	8%	Moderate
Our System	**XDP + ML**	**5M+ pps**	**95%+**	**<2%**	**Moderate**

2.7 Research Gap Analysis

Table 2.2: Research Gap Analysis

Gap	Existing Solutions	Our Contribution
G1: Speed vs Intelligence	Fast systems lack ML intelligence OR smart systems are slow	Hybrid eBPF/XDP (speed) + ML (intelligence)
G2: False Positives	Rule-based: 10-15% FP, ML-only: 3-5% FP	Hybrid statistical + ML: <2% FP
G3: Deployment Complexity	DPDK complex, cloud scrubbing adds latency/cost	Kernel-integrated eBPF, standard Linux
G4: Real-Time Adaptation	Static rules, slow ML retraining	Adaptive baseline, online model updates
G5: Comprehensive Evaluation	Limited comparisons, single metrics	Multi-metric evaluation, multiple baselines
G6: Practical Implementation	Theoretical or proprietary	Open-source, documented, reproducible

Summary of Research Gap

Primary Gap: No existing system comprehensively integrates eBPF/XDP kernel-level filtering with ML classification while achieving both high throughput (5M+ pps) and high accuracy (95%+) with low false positives (<2%).

Our Novel Contribution: Hybrid architecture seamlessly bridging kernel and user space for optimal performance and intelligence, with practical implementation and comprehensive evaluation.

[Continued in FINAL_REPORT_PART2.md with Chapters 3-8...]