

MSc in Cybersecurity & Digital Forensics

Masters Project Presentation

Shafee Ul Mahmud Chowdhury

Student ID: 2436374

University of Wales Trinity Saint David, London

Supervisor: Renuga Jayakumar

September 2026

Presenting on

**Adaptive AI-Based Real-Time Auto DDoS and Zero-Day
Detection and Mitigation for SDN Network**

Introduction

Traditional networking

Control and forwarding are distributed across network devices, making network-wide monitoring and coordinated security changes less direct. [2]

SDN separation

SDN separates the control plane from the data plane and enables programmable network control through a controller. [2]

Security opportunity

The controller can collect flow information and dynamically install forwarding or mitigation rules. [2]

- Data plane: OVS forwards packets and enforces flow entries.
- Control plane: OS-Ken processes traffic information and makes security decisions.
- OpenFlow provides the control path between the controller and switch. [3]

Introduction-Continue

Real-time DDoS detection

Detect malicious flow behavior using live SDN statistics and a trained Random Forest classifier [5]

Previously unseen behavior

Use behavioral analysis and Isolation Forest to identify traffic that differs from established benign behavior.

Automatic mitigation

Use OS-Ken and OpenFlow to translate the security decision into an OVS DROP rule

- The project evaluates the complete security loop rather than ML accuracy alone.
- Focus: detection → decision → mitigation → verification.

Literature Review

- DDoS attacks target availability by generating excessive or abnormal traffic that can consume bandwidth and network and service resources [1]
- DDoS is suitable for flow-level analysis because packet rate, byte rate, flow counts and connection behavior change during an attack . [6]
- SDN provides a programmable point for collecting these flow signals and applying mitigation dynamically [2]

Observable impact

Latency and response
degradation

Observable signals

Packets • bytes • rates •
flows

Research response

Detect → decide → block
[5]

Research Gap

Gap 1 — Benchmark dependence

Existing studies often rely on benchmark datasets; their traffic distribution may not fully represent live conditions. [4]

Gap 2 — Generalization

Previously unseen behavior is not always evaluated consistently, and 'zero-day' may be defined differently across studies. [8]

Gap 3 — End-to-end evidence

Classification metrics are often stronger than operational evaluation of detection latency, mitigation response and service impact [5]

- This project therefore combines supervised classification ,behavioral detection , automated mitigation and network-performance evaluation [5], [8]
- The unseen attack is treated as a controlled experimental condition, not as proof of universal real-world zero-day detection.

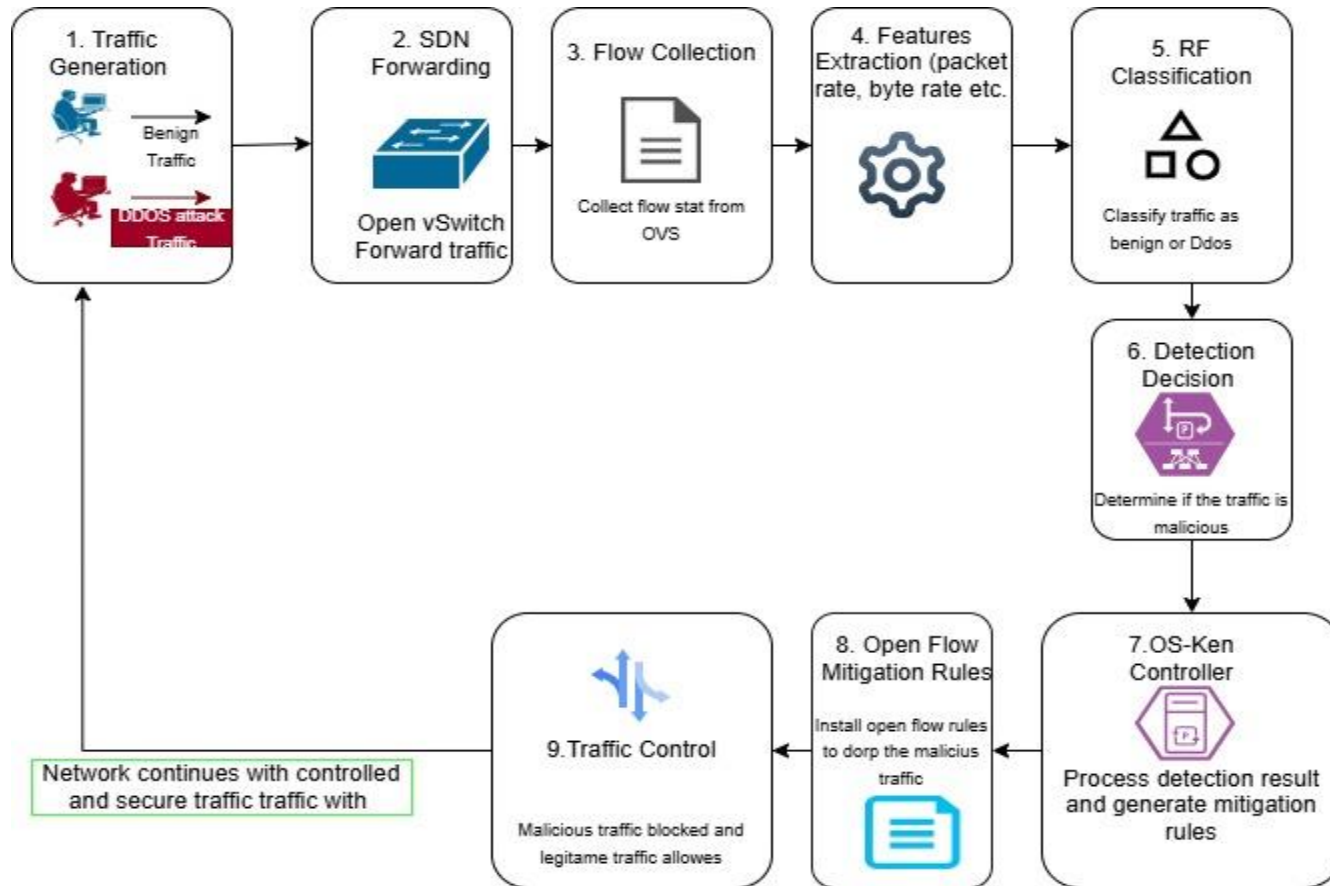
Aim and Objectives

Aim

Develop and evaluate an adaptive AI-based real-time DDoS detection and mitigation framework in a controlled SDN environment.

- Build the Mininet , OVS and OS-Ken SDN testbed.
- Use InSDN-derived features for supervised DDoS classification .
- Combine Random Forest with behavioral detection.
- Automate OpenFlow mitigation through OS-Ken.
- Evaluate classification, detection behavior, mitigation and network performance.

Methodology



1. Observe

Collect live flow statistics from the SDN/OVS environment.

2. Classify

Create the same 20-feature representation used by the trained RF model

3. Decide + mitigate

RF + behavioral logic → OS-Ken
→ OpenFlow DROP → OVS

SDN Environment-Core Component

OS-Ken

Python-based controller framework used for traffic analysis, security logic and OpenFlow enforcement .

OpenFlow 1.3

Communication/control protocol used to install, modify and remove flow entries . [3]

Open vSwitch

Software data-plane switch used to forward traffic and enforce DROP rules . [3]

Mininet

Controlled virtual SDN testbed.

- OS-Ken was selected after practical compatibility issues with the initially considered Ryu setup.[3]
- Python provides a common environment for controller logic, preprocessing and ML inference.

SDN Environment-Core component

AI

Supports pattern recognition and automated security decisions.

ML

Learns relationships among multiple flow features instead of relying on a single threshold . [6]

Random Forest

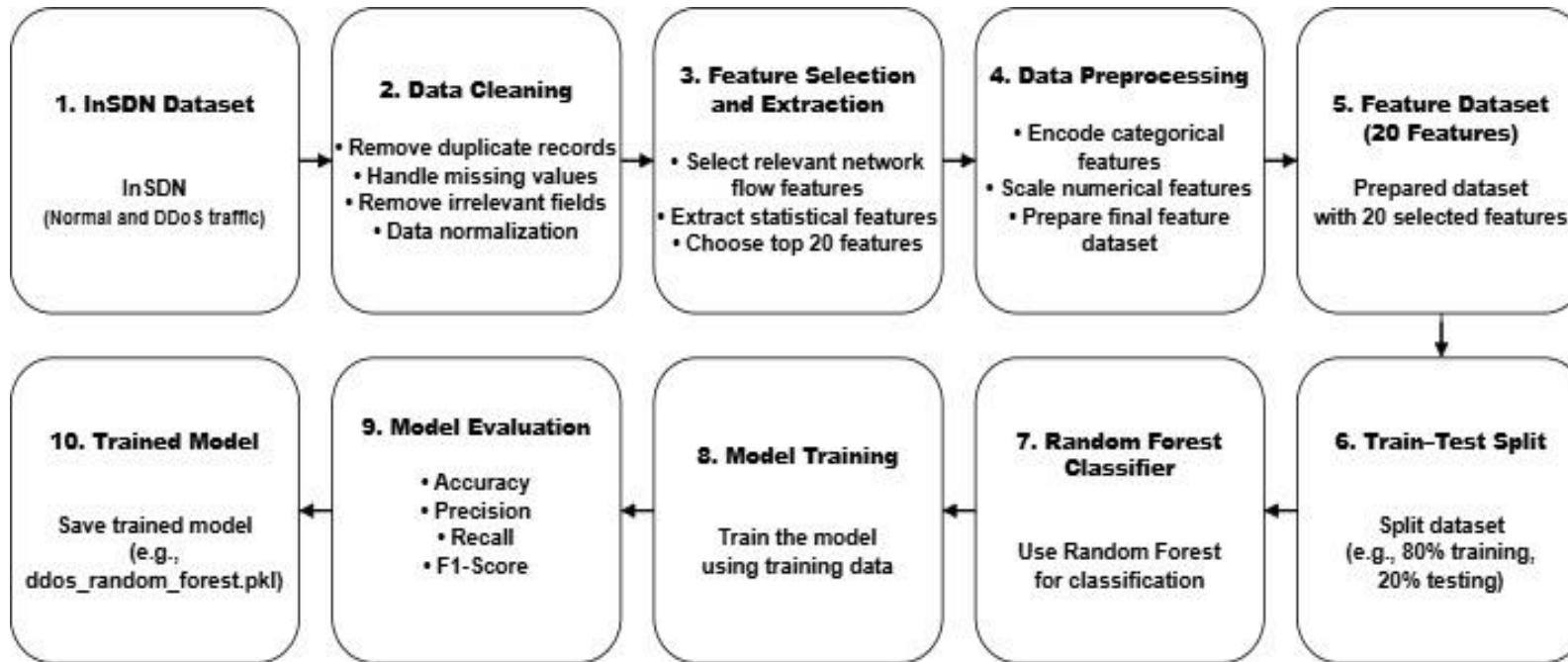
Works well with structured flow features and provides practical classification/feature analysis . [9]

Python

Connects data processing, Scikit-learn and the OS-Ken controller.

- RF was preferred over a more complex deep-learning model because the research evaluates an end-to-end pipeline where inference practicality and response time also matter .
- Python libraries support preprocessing, training, inference and evaluation.

Inside of Dataset



Selected 20 Features from InSDN Dataset (Examples)

- | | | | |
|---------------------|-----------------------------------|------------------------------|---------------------------------|
| 1. Protocol | 6. Packet Rate | 11. Source Flow Count | 16. Destination Byte Count |
| 2. Source Port | 7. Byte Rate | 12. Destination Flow Count | 17. Forward Packet Length Mean |
| 3. Destination Port | 8. Flow Duration | 13. Source Packet Count | 18. Backward Packet Length Mean |
| 4. Total Packets | 9. Average Packet Size | 14. Destination Packet Count | 19. Flow Flags |
| 5. Total Bytes | 10. Flow IAT (Inter-arrival Time) | 15. Source Byte Count | 20. Label (Normal / DDoS) |

Reference dataset

InSDN was used as the principal SDN intrusion dataset.

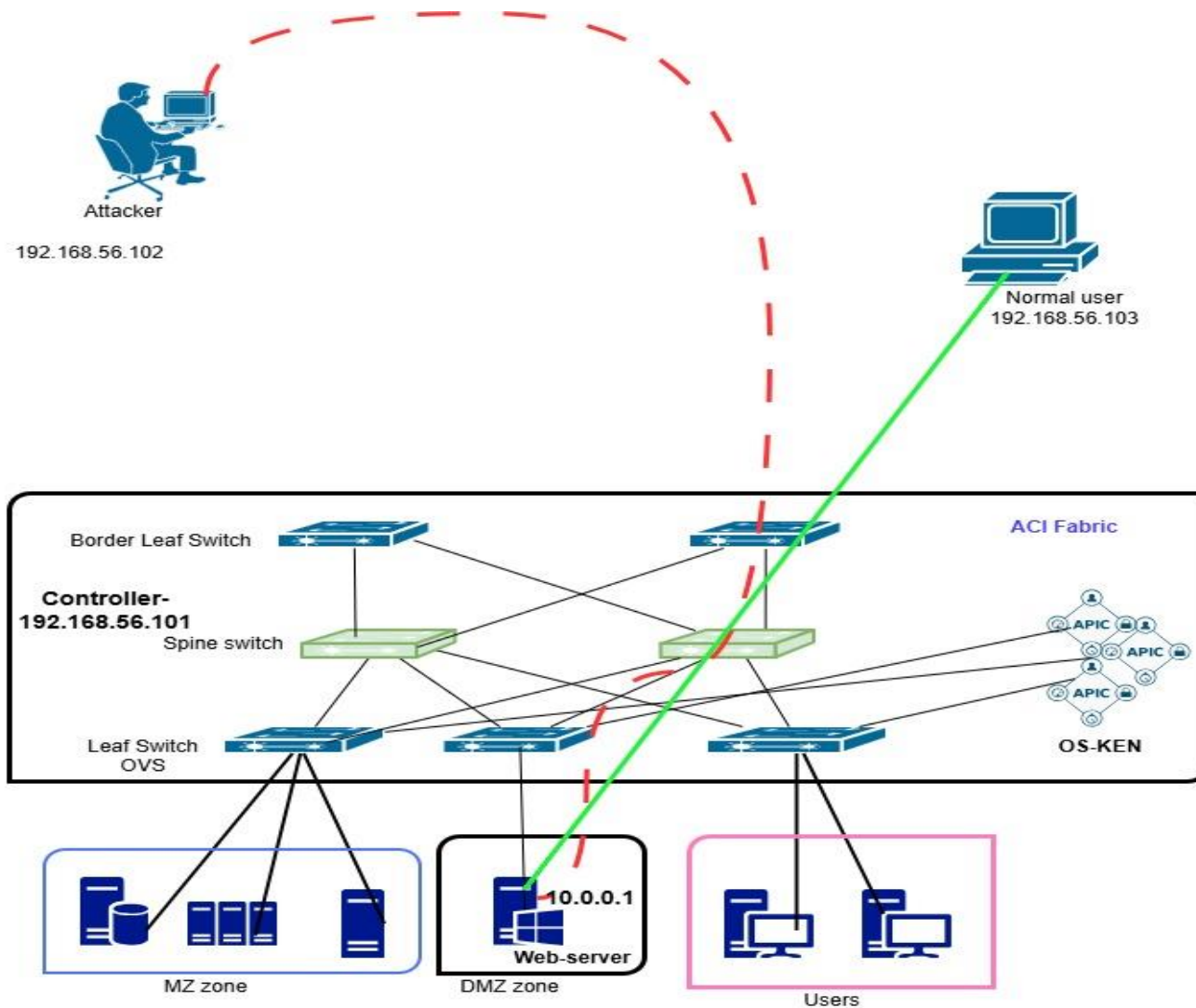
Project data

37,079 records; 20 features;
13,954 BENIGN and 23,125 DDoS.

Learning task

Binary classification:
BENIGN vs DDoS .

Structure of Lab environment



Kali / attacker

192.168.56.102

Protected web service

10.0.0.1

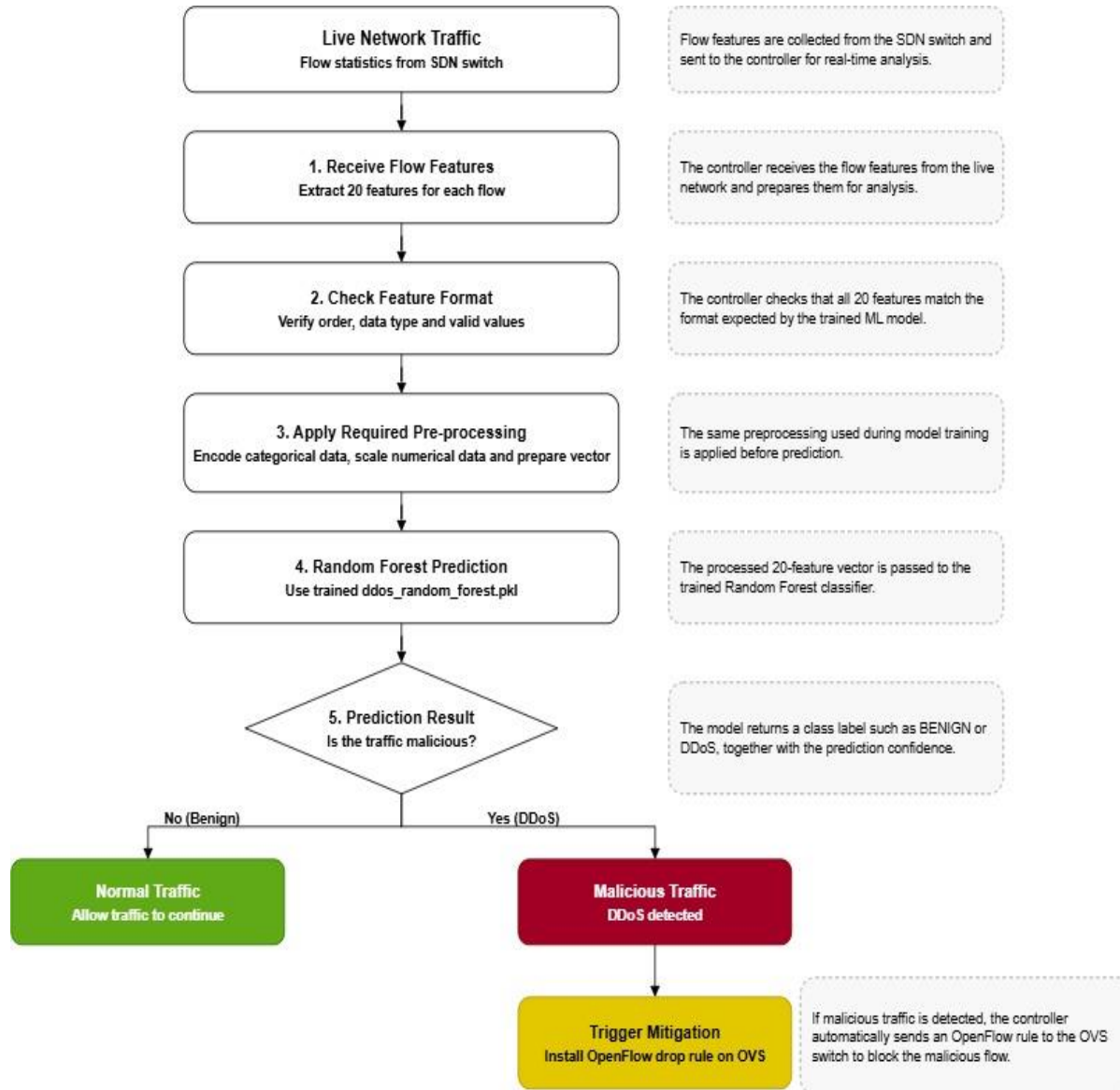
OS-Ken controller

192.168.56.101

Normal user

192.168.56.103

Traffic Classification-Flow



- OVS flow/packet statistics are collected
- 20 traffic and behavioral features are prepared
- Random Forest predicts BENIGN or DDoS
- Behavioral logic provides an additional signal.
- BENIGN → forward normally; DDoS/abnormal → OS-Ken security logic → mitigation .

Random Forest Validation

```
(osken-env) root@Ubuntu-SDN:/home/safi/SDN-Project# python3 validate_model.py
```

```
=====
MODEL VALIDATION
=====
```

```
Records: 37079
Features: 20
BENIGN: 13954
DDoS: 23125
```

```
Existing Random Forest loaded successfully.
```

```
=====
5-FOLD CROSS VALIDATION
=====
```

```
-----
CROSS-VALIDATION PERFORMANCE
-----
```

```
Accuracy  : 91.78% +/- 0.23%
Precision  : 91.24% +/- 0.25%
Recall     : 91.27% +/- 0.27%
F1-Score   : 91.25% +/- 0.24%
```

```
-----
FOLD RESULTS
-----
```

```
Fold 1: Accuracy=91.86% | Precision=91.20% | Recall=91.53% | F1=91.36%
Fold 2: Accuracy=92.08% | Precision=91.57% | Recall=91.57% | F1=91.57%
Fold 3: Accuracy=91.92% | Precision=91.45% | Recall=91.32% | F1=91.38%
Fold 4: Accuracy=91.60% | Precision=91.09% | Recall=90.99% | F1=91.04%
```

Accuracy

91.78%

Precision / Recall

91.24%/ 91.27%

F1-score

91.27%

Validation

Normal Behavior

```
CONFIDENCE      : 1.0
METHOD          : RF+BASELINE
ANOMALY SCORE   : 0
REASON          : normal behavior
=====

SOURCE IP       : 10.0.0.1
DESTINATION IP  : 192.168.56.103
PROTOCOL        : ICMP
SOURCE PORT     : 0
DESTINATION PORT : 0

PACKETS         : 51
BYTES           : 4998
PACKET RATE     : 0.996 packets/sec
BYTE RATE       : 97.597 bytes/sec
DURATION        : 51.211 seconds
SOURCE FLOW COUNT : 1
SOURCE RATE     : 0.996 packets/sec
DEST FLOW COUNT  : 1
DESTINATION RATE : 0.996 packets/sec
=====

HYBRID DDoS DETECTION
=====

CLASS           : BENIGN
CONFIDENCE      : 1.0
METHOD          : RF+BASELINE
ANOMALY SCORE   : 0
REASON          : normal behavior
=====
```

Classification

BENIGN

Confidence

1.0

Decision

Continue normal forwarding; no mitigation rule

Abnormal Behavior

```
DESTINATION IP : 10.0.0.2
PROTOCOL       : ICMP
SOURCE PORT    : 0
DESTINATION PORT : 0

PACKETS       : 1340
BYTES         : 131320
PACKET RATE   : 111.372 packets/sec
BYTE RATE     : 10914.477 bytes/sec
DURATION      : 12.032 seconds
SOURCE FLOW COUNT: 1
SOURCE RATE    : 111.372 packets/sec
DEST FLOW COUNT : 1
DESTINATION RATE : 111.372 packets/sec

=====
HYBRID DDoS DETECTION
=====
CLASS      : DDoS
CONFIDENCE : 0.99
METHOD     : BEHAVIOR
ANOMALY SCORE : 13
REASON      : packet_rate above INSDN baseline; byte_rate above INSDN baseline; ICMP flood rate; high packet-rate burst; high source packet rate; above adaptive live baseline; source rate above adaptive baseline; destination rate above adaptive baseline

!!! DDoS / ABNORMAL TRAFFIC DETECTED !!!
!!! AUTO MITIGATION ACTIVE !!!

=====
[DROP COUNTER ACTIVE] 10.0.0.2 -> 10.0.0.1 | packets: 3069 | bytes: 300762
[MITIGATION ACTIVE] 10.0.0.2 -> 10.0.0.1 | DROP packets: 3069 | DROP bytes: 300762 | RULE KEPT
[
```

Classification

DDoS

Confidence

0.99

Behavior evidence

Anomaly score 13; elevated packet rate and burst indicators

Interpretation

Controlled unseen-behaviour test.

Rule Implementation

```
(osken-env) root@Ubuntu-SDN:/home/safi/SDN-Project# sudo ovs-ofctl -O OpenFlow13 dump-flows s1 | grep "cookie=0xd00d"
cookie=0xd00d, duration=74.890s, table=0, n_packets=318674, n_bytes=17208396, idle_timeout=30, priority=500,ip,nw_src=192.168.56.102,nw_dst=10.0.0.1 actions=drop
(osken-env) root@Ubuntu-SDN:/home/safi/SDN-Project#
```

Mitigation rule

Cookie: 0xd00d

Priority: 500

Match

192.168.56.102 → 10.0.0.1

Action

DROP

Evidence

Packet/byte counters

Service interruption

```
64 bytes from 10.0.0.1: icmp_seq=71 ttl=63 time=5.33 ms
64 bytes from 10.0.0.1: icmp_seq=72 ttl=63 time=3.13 ms
64 bytes from 10.0.0.1: icmp_seq=73 ttl=63 time=2.67 ms
64 bytes from 10.0.0.1: icmp_seq=74 ttl=63 time=4.14 ms
64 bytes from 10.0.0.1: icmp_seq=75 ttl=63 time=2.18 ms
64 bytes from 10.0.0.1: icmp_seq=76 ttl=63 time=384 ms
64 bytes from 10.0.0.1: icmp_seq=82 ttl=63 time=2257 ms
64 bytes from 10.0.0.1: icmp_seq=89 ttl=63 time=2214 ms
64 bytes from 10.0.0.1: icmp_seq=92 ttl=63 time=2486 ms
64 bytes from 10.0.0.1: icmp_seq=94 ttl=63 time=2232 ms
64 bytes from 10.0.0.1: icmp_seq=102 ttl=63 time=2091 ms
64 bytes from 10.0.0.1: icmp_seq=104 ttl=63 time=2007 ms
64 bytes from 10.0.0.1: icmp_seq=129 ttl=63 time=103 ms
64 bytes from 10.0.0.1: icmp_seq=130 ttl=63 time=397 ms
64 bytes from 10.0.0.1: icmp_seq=131 ttl=63 time=3.78 ms
64 bytes from 10.0.0.1: icmp_seq=132 ttl=63 time=7.30 ms
64 bytes from 10.0.0.1: icmp_seq=133 ttl=63 time=3.05 ms
64 bytes from 10.0.0.1: icmp_seq=134 ttl=63 time=4.42 ms
64 bytes from 10.0.0.1: icmp_seq=135 ttl=63 time=3.99 ms
64 bytes from 10.0.0.1: icmp_seq=136 ttl=63 time=3.39 ms
64 bytes from 10.0.0.1: icmp_seq=137 ttl=63 time=2.94 ms
```

- Observed values include low-latency normal operation and increased response during attack conditions
- Latency evidence was used alongside controller logs and OVS flow counters rather than as a standalone security metric

Evaluation

```
64 bytes from 10.0.0.1: icmp_seq=71 ttl=63 time=5.33 ms
64 bytes from 10.0.0.1: icmp_seq=72 ttl=63 time=3.13 ms
64 bytes from 10.0.0.1: icmp_seq=73 ttl=63 time=2.67 ms
64 bytes from 10.0.0.1: icmp_seq=74 ttl=63 time=4.14 ms
64 bytes from 10.0.0.1: icmp_seq=75 ttl=63 time=2.18 ms
64 bytes from 10.0.0.1: icmp_seq=76 ttl=63 time=384 ms
64 bytes from 10.0.0.1: icmp_seq=82 ttl=63 time=2257 ms
64 bytes from 10.0.0.1: icmp_seq=89 ttl=63 time=2214 ms
64 bytes from 10.0.0.1: icmp_seq=92 ttl=63 time=2486 ms
64 bytes from 10.0.0.1: icmp_seq=94 ttl=63 time=2232 ms
64 bytes from 10.0.0.1: icmp_seq=102 ttl=63 time=2091 ms
64 bytes from 10.0.0.1: icmp_seq=104 ttl=63 time=2007 ms
64 bytes from 10.0.0.1: icmp_seq=129 ttl=63 time=103 ms
64 bytes from 10.0.0.1: icmp_seq=130 ttl=63 time=397 ms
64 bytes from 10.0.0.1: icmp_seq=131 ttl=63 time=3.78 ms
64 bytes from 10.0.0.1: icmp_seq=132 ttl=63 time=7.30 ms
64 bytes from 10.0.0.1: icmp_seq=133 ttl=63 time=3.05 ms
64 bytes from 10.0.0.1: icmp_seq=134 ttl=63 time=4.42 ms
64 bytes from 10.0.0.1: icmp_seq=135 ttl=63 time=3.99 ms
64 bytes from 10.0.0.1: icmp_seq=136 ttl=63 time=3.39 ms
64 bytes from 10.0.0.1: icmp_seq=137 ttl=63 time=2.94 ms
```

Normal

≈ 2.18 ms RTT

During DDoS

≈ 2091 ms RTT

Post-mitigation

≈ 2.94 ms RTT

Observation

Service connectivity
recovered after mitigation

Evaluation

Accuracy

91.96%

Precision

93.37%

Recall

93.77%

F1-score

93.57%

- ML: accuracy, precision, recall and F1-score
- Robustness: controlled unseen-behavior experiment
- Operational evidence: detection output, OS-Ken decision, OVS rule installation and counters
- Network evidence: normal vs attack vs post-mitigation latency and response behavior.

Evaluation

Accuracy

91.78%

Precision

91.24%

Recall

91.27%

F1-score

91.25%

```
(osken-env) root@Ubuntu-SDN:/home/safi/SDN-Project# python3 validate_model.py
```

```
=====
MODEL VALIDATION
=====
```

```
Records: 37079
Features: 20
BENIGN: 13954
DDoS: 23125
```

```
Existing Random Forest loaded successfully.
```

```
=====
5-FOLD CROSS VALIDATION
=====
```

```
-----
CROSS-VALIDATION PERFORMANCE
-----
```

```
Accuracy   : 91.78% +/- 0.23%
Precision   : 91.24% +/- 0.25%
Recall      : 91.27% +/- 0.27%
F1-Score    : 91.25% +/- 0.24%
-----
```

Comparison

Comparison	Existing approach	Proposed Project
1. Detection approach	Traditional security mechanisms can depend on predefined rules, signatures or manually configured policies[4]	Uses Random Forest + behavioral analysis + Isolation Forest to analyze flow behavior and detect DDoS or previously unseen abnormal behavior
2. Response and Mitigation	Detection separate from the mitigation process, requiring security policies or network changes to be applied.[5]	Detection → OS-Ken decision → OpenFlow DROP rule → OVS mitigation is integrated into one pipeline

Limitations

- Evaluation is based on a controlled Mininet/OVS environment.
- Larger and more diverse environments would strengthen confidence in zero-day detection.
- Behavioral thresholds and baseline statistics depend on the experimental traffic environment.

Production-scale controller throughput and scalability were not evaluated.

Future Work

- Evaluate larger and more diverse SDN topologies and higher traffic volumes .
- Test additional and lower-rate DDoS patterns .
- Measure controller CPU, memory, detection latency and mitigation latency .
- Compare RF with alternative models under the same experimental protocol .

Conclusion

Detection

Random Forest provides the supervised BENIGN/DDoS classification layer, supported by behavioral analysis

Mitigation

OS-Ken converts the security decision into OpenFlow rules enforced by OVS

Verification

OVS counters and network-performance observations provide evidence of mitigation and service recovery

- The project demonstrates a complete SDN security loop: observation → classification → decision → mitigation → verification
- The contribution is the integration and evaluation of supervised detection, behavioral novelty analysis and programmable mitigation
- Results remain bounded by the dataset and controlled experimental environment

References

- [1] S. T. Zargar, J. B. Joshi, and D. Tipper, “A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013, doi: 10.1109/SURV.2013.031413.00127.
- [2] D. Kreutz, F. M. V. Ramos, P. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015
- [3] H. Riggs, A. Khalid, and A. I. Sarwat, “An overview of SDN issues—A case study and performance evaluation of a secure OpenFlow protocol implementation,” *Electronics*, vol. 14, no. 16, Art. no. 3244, Aug. 2025,
- [4] W. Hill, Y. T. Acquaaah, J. Mason, D. Limbrick, S. Teixeira-Poit, C. Coates, and K. Roy, “DDoS in SDN: A review of open datasets, attack vectors and mitigation strategies,” *Discover Applied Sciences*, vol. 6, Art. no. 472, 2024

References-continue

[5] O. Rahman, M. A. G. Quraishi, and C.-H. Lung, “DDoS attacks detection and mitigation in SDN using machine learning,” in Proc. IEEE World Congress on Services (SERVICES), Milan, Italy, 2019, pp. 184–189,

[6] D. Han, H. Li, X. Fu, and S. Zhou, “Traffic feature selection and distributed denial of service attack detection in software-defined networks based on machine learning,” Sensors, vol. 24, no. 13, Art. no. 4344, 2024,

[7] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP), 2018, pp. 108–116,

[8] A. Hirsi, L. Audah, A. Salh, N. M. Sahar, S. Ahmed, and M. A. Alhartomi, “DDoS anomaly detection in software-defined networks: An evaluation of machine learning techniques for traffic classification and prediction,” in Proc. 2024 Int. Conf. on Future Technologies for Smart Society (ICFTSS), 2024,

[9] L. Breiman, “Random forests,” Machine Learning, vol. 45, pp. 5–32, 2001,

Thank You