SDN Intrusion Detection with Machine Learning and Real-Time Simulation

Othmani Soumaya Hamdi Mohamed Amine Chaouch Cherifa
soumaya.othmani@insat.ucar.tn *mohamedamine.hamdi@insat.ucar.tn* *cherifa.chaouch@insat.ucar.tn*

Gritli Skander Ben Salem Meryem
skander.gritli@insat.ucar.tn *meryem.bensalem@insat.ucar.tn*

***Abstract*—Software-Defined Networking, or SDN, has emerged as a revolutionizing and transformative technology in network architecture redefining conventional approaches with a new level of programmability and adaptability and offering tremendous flexibility and efficiency. These advantages, however, come with increased exposure to security vulnerabilities, making the SDN environment more susceptible to various kinds of cyber-attacks. Intrusion Detection Systems based on Machine Learning have shown great promise in effectively detecting and mitigating these threats, especially in the modern and sophisticated infrastructures of SDNs. In such a context, this research introduces an Intrusion Detection in a real-time framework that not only considers state-of-the-art ML-based techniques for anomaly detection but also integrates a real-time attack simulation environment that may allow dynamic assessment of IDS performance against various emulated attack scenarios, thus offering a truly realistic and comprehensive evaluation of their effectiveness and robustness.**

***Index Terms*—SDN, Machine Learning, Intrusion Detection System, Real-time simulation, NSL-KDD, Network Security**

I. Introduction

In the era of ubiquitous web applications, strong network security assurances have become the foremost of concerns. Software Defined Networking represents a new, programmable approach to managing high-bandwidth and dynamic networks with flexibility and scalability. However, on the other hand, its increased flexibility is what makes it more exposed to a broader attack range and thus more vulnerable to numerous cyber threats. Advanced and adaptive security concepts are imperative to protect the SDN environments from such evolvingly critical threats.

Intrusion Detection Systems are very instrumental in the identification and detection of adverse activities over a network. The signature-based systems among other traditional IDS approaches efficiently detect the known attacks by matching the network traffic against predefined attack signatures. While these systems do well for the known threats whose documentation is complete, they display inefficiency in detecting unknown or zero-day attacks that have not been previously cataloged in their signature databases. In contrast, anomaly-based IDSs use ML models in an effort to transcend this limitation by identifying deviations from established normative behavior patterns. Such systems are capable of detecting attacks that were previously unknown, which makes them particularly important in dynamic environments.

Despite these merits, most of the IDS frameworks are still devoid of integration with dynamic testing environments. In fact, this seriously diminishes their practical applicability, as such frameworks are not able to dynamically test the performance of IDS models against a variety of simulated, real-world attack scenarios. Without such testing frameworks, most of the proposed IDS solutions remain untested under the complex conditions of live networks, where threats can vary drastically in their nature and severity.

This paper describes a new IDS framework adapted to SDN using ML-based anomaly detection in combination with an integrated real-time simulation environment. The system is intended for the testing of IDS robustness by running realistic attack scenarios and generating real-time network traffic, coupled with adversarial behaviors that could create dynamic environments of threats. This would allow real-time testing that gives great insights into how effective the IDS will be under various network conditions and attack types. It enables the system to update itself for an effective response against dynamically changing threats, improving the efficiency in detecting both known and unknown attacks in the SDN environment.

The IDS framework proposes ML-based anomaly detection methods, including clustering-based feature selection, dimensionality reduction approaches such as PCA, and efficient classifiers like Random Forest, Extra Trees, and LightGBM. These techniques enable the system to sift out effectively the anomalies from normal behavior; hence, high detection and lower false alarm rates have been guaranteed. Moreover, the existence of a real-time simulator makes the system more appropriate for real-world use by network administrators who want to get sufficient knowledge about IDS behavior in real attack situations.

Empirical evaluations of the proposed approach using NSL-KDD dataset manifests positive results in showing its efficiency. It follows that the framework of IDS outperforms other solutions in the areas of accuracy, precision, recall, and F-measure. The same evidence proves that this system can enhance the security of SDN through the depth of the detection methodology to tackle old and new cyber-attacks.

To conclude, the proposed IDS framework is a high-performance, adaptable and real-time solution for SDN security. Machine learning coupled with a responsive simulation environment enforces protection efficiently against all types malicious activities and hence can act as a promising tool for safeguarding any SDN in the future.

II. Background

A. *Software-Defined Networking (SDN)*

Software Defined Networking (SDN) is a dynamic topology that is programmable, low-cost, and flexible. The network control and forwarding responsibilities are logically isolated, by which network control is readily programmable while the underlying infrastructure for applications and network services is abstracted.

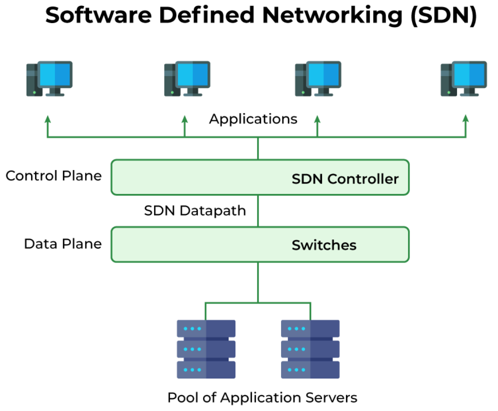


Fig. 1. Software Defined Networking Architecture

There are several models commonly used in Software Defined Networking, each provides certain advantages that are suited for different use cases. Examples include **Open SDN**, which separates the control plane from the data plane by using open protocols such as OpenFlow; SDN via APIs, whereby application programming interfaces allow communication and integration between network devices and controllers; **SDN via Hypervisor-based Overlay Networks**, which depend on virtualization to create overlay networks for traffic management; and **Hybrid SDN**, where traditional networking is combined with other approaches to ensure seamless transition and compatibility between old and new technologies.

As mentioned earlier, SDN key concept is that it seperates the network's control and data planes, enabling centralized management and programmability. This architecture comprises three main components:

- **SDN Applications**: Interface with network controllers for user interaction and policy management.
- **SDN Controllers**: Act as the network's *brain*, gathering data from devices and dispatching instructions.
- **SDN Networking Devices**: Execute data forwarding and processing as directed by controllers.

SDNs are being exploited in various network applications, from residential and corporate networks to data centers. It is crucial in modern networking because of its manifold advantages. It enhances network connectivity, thus making sales, services, and internal communications smoother, and allows data to be shared at a faster speed. SDN also facilitates application deployment, allowing new applications and business models to be implemented really quickly. Morover, on the side of security, it provides high visibility across the network offering the ability of dynamic responses like the isolation of devices according to security requirements. It is undeniable that SDN gives much better control and traffic management at a very high pace using an open-standard software-based controller and is thereby superior compared to traditional forms of networking.

However, despite its benefits, SDN is prone to unique vulnerabilities such as Distributed Denial of Service (DDoS) attacks, spoofing, and controller-targeted exploits.

B. *Intrusion Detection System IDS*

In the domain of network security, Intrusion Detection Systems are an integral component in detecting malicious activities, abnormalities, and potential threats. IDS implementations in the context of SDN come uniquely equipped to leverage the programmability and flexibility features of SDN. IDS acts as a monitoring mechanism that is used for continuous analysis of network traffic, using algorithms to identify patterns indicative of unauthorized access. Traditional approaches to IDS, such as signature-based detection, are quite challenging in SDN due to the frequently changing topology and large traffic volume. Due to such a reason, Machine Learning-based solutions for IDS have become popular over time in the field of SDN.

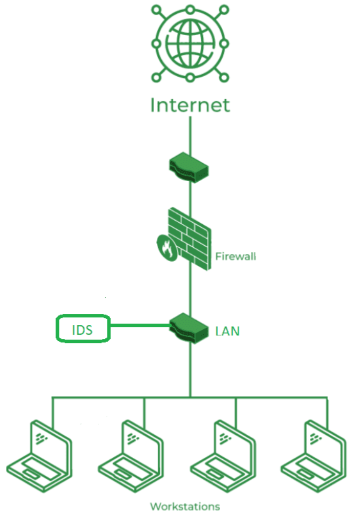


Fig. 2. Intrusion Detection System (IDS)

C. *Machine Learning in IDS*

Machine Learning has emerged as a game-changer in the field of network security by equipping it with powerful tools for anomaly detection that can analyze behavioral patterns and differentiate between normal and malicious network activities, all with unprecedented speed and accuracy. ML-based IDS utilizes data-driven models in detecting patterns and anomalies, overcoming some of the traditional methods in finding novel and sophisticated cyber-attacks. By analyzing vast amounts of traffic data, ML algorithms classify activities as either benign or malicious and increase the effectiveness of IDS solutions by a great extent. Key ML techniques for IDS include:

- **Hybrid Feature Selection (HFS)**: Feature selection is a very crucial step for any ML model in performance optimization. Hybrid Feature Selection embeds different methods such as Random Forest Recursive Feature Elimination (RF-RFE) and Correlation-Based Feature Selection (CFS) to choose the best features while removing redundancy. This results in a concise but informative feature set that enhances the efficiency and accuracy of ML models.
- **Classifiers**:Most of the used classification algorithms are LightGBM and Random Forest (RF) since they have a lot of power in handling imbalanced datasets with much ease. These models have been very efficient in the detection of attacks by prioritizing the minority classes and ensuring high accuracy even in the skewed dataset. Their adaptability makes them very suitable for dynamic, large-scale network environments.

Machine learning techniques span from supervised to unsupervised learning methods. Supervised learning methods utilize labeled datasets to train models that identify known attack patterns, while unsupervised methods based on clustering are very effective for anomaly detection. This enables a combination of such techniques that allows IDSs themselves to adapt to evolving attack patterns and emerging threats. Recent developments in deep learning and hybrid machine learning models have further enhanced IDS with deep architectures that can effectively learn complex patterns in network traffic. This, therefore, provides an enhanced potential to detect both known and zero-day threats.

ML algorithms have been shown to be effective in a large number of IDS applications. Singh et al. presented an ML-based defense mechanism against Distributed Denial-of-Service (DDoS) attacks in SDN. The system, which was divided into modules for the collection of flow statistics, feature extraction, training, and traffic classification, evaluated several classifiers, including Support Vector Machine (SVM), Decision Tree (DT), Logistic Regression (LR), and Nearest Neighbors (k-NN). The classifier with the highest accuracy, along with the lowest false positive rate, was selected for deployment. Similarly, Chen et al. proposed an attack detection method using the XGBoost classifier. Their system integrated a traffic collector and a classification model inside the controller, through the special characteristics of the SDN controller. This approach efficiently avoids threats targeted at the SDN controller, which is a critical component in the network architecture.

III. Limitations of existing works

What makes machine learning–based technologies inevitable parts of modern IDS solutions is their flexibility. Thus, they are capable of finding hidden patterns in network traffic to make a clear distinction between legitimate and malicious one. Moreover, the ML models provide resilient mechanisms for the detection and prevention of various SDN threats, including DDoS attacks and APTs. Still, challenges persist. Most of the machine learning-based intrusion detection systems rely on outdated or imbalanced datasets to train their models, which may limit their practical applications. Advanced data preprocessing techniques, such as normalization, feature resampling, and dimensionality reduction, are necessary to ensure that models work well in real-world scenarios. Furthermore, the balance between security and privacy is a major concern, as most of the machine learning algorithms are usually trained on sensitive data. Machine Learning has been the backbone of modern IDS solutions, bringing unparalleled capabilities of intrusion detection and mitigation over the networks. Combining advanced feature selection methods, classifiers, and supervised and unsupervised learning techniques, the ML-driven IDS system adjusts to the dynamic nature of cyber threats. The fact that these ML algorithms continue to evolve and that novel approaches keep being developed in order to deal with challenges related to data ensures that they are still relevant today for network security.

IV. Methodology

The proposed methodology for this research will be base on developing an effective and resilient Machine Learning-based Intrusion Detection System capable of handling large imbalanced datasets. The paper henceforth outlines the methodological steps we took regarding dataset selection,data preprocessing, feature selection, and model training and evaluation .

A. Dataset Selection and Preprocessing

The NSL-KDD dataset,a standard benchmark for IDS evaluation, serves as the basis for this study. It contains labeled samples representing various attack types, including Denial of Service (DoS), Probe, User-to-Root (U2R), and Remote to Local (R2L). The following steps are taken to preprocess the dataset to make it suitable for training the ML model.

- **Data Cleaning**: Duplicate entries and missing values were removed to ensure the integrity of the data. Outliers were detected and handled to minimize their effect on model performance.
- **Feature Scaling**: Numerical features were standardized to have zero mean and unit variance, ensuring uniformity across feature scales. Categorical features were encoded using one-hot encoding to make them compatible with ML algorithms.
- **Class Imbalance Handling**: The class imbalance inherent in the dataset was tackled using Random Oversampling of the minority class instances, thus giving them equal representation in the model, which improved its ability to detect less frequent attacks.

- **Exploratory Data Analysis (EDA)**: The distribution of each feature and their inter-correlation were examined to inform feature selection and dimensionality reduction steps later on.

B. Hybrid Feature Selection (HFS)

In this view, a Hybrid Feature Selection (HFS) strategy was implemented to enhance the efficiency and predictive abilities of the system deploying two complementary techniques, RF-RFE and CF. The HFS process utilized RF-RFE and CFS to remove redundant and irrelevant features, ensuring the dataset's dimensionality was optimized. This step improved classification accuracy and reduced computational overhead. RF-RFE iteratively eliminated the least important features based on their contribution to model performance, while CFS identified characteristics that were marked by strong predictive relevance and contained low inter-correlation and prioritized features accordingly.

C. ML Models and Training

our proposed framework designs and evaluates two separate machine learning models, each one is specialized in dealing with specific aspects of network intrusion detection.

- **Model 1: HFS-LGBM Approach**
  - Binary classification (attack vs. normal traffic):
    The model aims at classifying network traffic into either the normal or attack classes, offering a very high accuracy and low false positive rate in the detection phase.

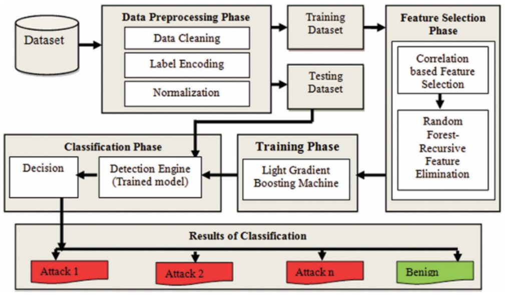
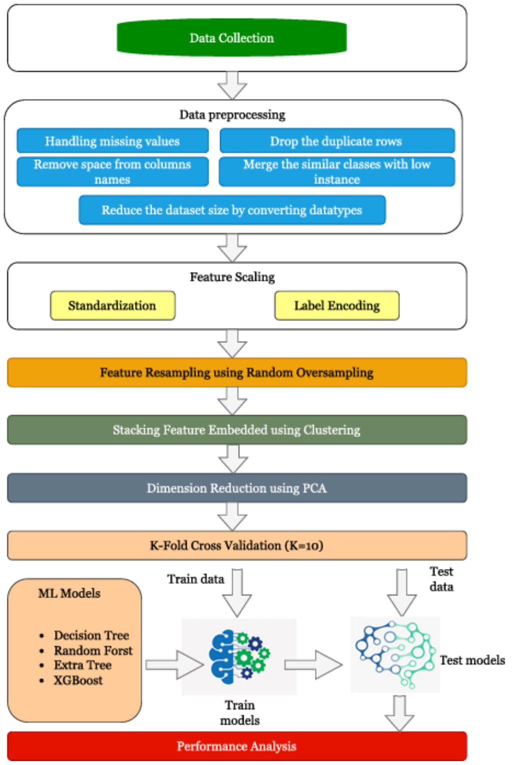


Fig. 3. HFS-LGBM pipeline

  - Light Gradient Boosting Machine (LightGBM) usage:
    LightGBM is a tree-based boosting algorithm known for having very high computational efficiency in term of speed and memory and it also supports large-scale datasets. It uses histogram-based learning for memory usage optimization and Hyperparameter Tuning like the number of boosting iterations, learning rate, and maximum depth for model's performance optimization.
  - Other Models usage:
    In Addition to LightGBM , Other models such as SVM , Logistic Regression, K-Nearest Neighbors, Random Forest, Decision Tree, Gradient Boosting and XGBoost , were taken into consideration .
  - Optimized Feature Set:
    The feature set, optimized through Hybrid Feature Selection (HFS), has reduced noise and increased the interpretability of the models, so that the classifier can focus on the most predictive features.
- **Model 2: Network Intrusion Detection Pipeline** The current model was created for multiclass classification in order to distinguish various classes of attacks, such as DoS, Probe, U2R, and R2L.



Data Collection
Data preprocessing
Handling missing values
Drop the duplicate rows
Remove space from columns names
Merge the similar classes with low instance
Reduce the dataset size by converting datatypes
Feature Scaling
Standardization
Label Encoding
Feature Resampling using Random Oversampling
Stacking Feature Embedded using Clustering
Dimension Reduction using PCA
K-Fold Cross Validation (K=10)
Train data
Test data
ML Models
• Decision Tree
• Random Forst
• Extra Tree
• XGBoost
Train models
Test models
Performance Analysis

Fig. 4. Network diagram of SDN with IDS integration

Sophisticated preprocessing methodologies are integrated into its design to ensure better accuracy and generality:

  - This model predicts malicious activities into specific categories of attacks, thus furnishing in-depth information regarding the features of the detected intrusion.
  - Dimensionality Reduction through Clustering:
    K-Means clustering algorithm is used to create meat features and group similar data points, which helped in identifying the natural clusters within the dataset
  - Principal Component Analysis (PCA):
    After clustering, PCA was applied to further reduce dimensionality while retaining as much variance as possible.

Both models were subjected to extensive training using 10-fold cross-validation to minimize overfitting and ascertain their reliability and practical utility.

Their synergistic emphasis on both binary and multiclass classification offers a holistic approach to intrusion detection within Software-Defined Networking environments. The findings of these models illustrate marked enhancements in detection precision, rates of false positives, and their ability to adapt to changing attack patterns.

V. Results and Discussion

A. Evaluation Metrics

The proposed models were evaluated on multiple performance metrics to ensure robustness and practical applicability:

- Accuracy: The proportion of correctly classified samples.
- Precision: The ratio of true positives to all positive predictions.
- Recall (Sensitivity): Number of true positives detected out of all the actual positive cases.
- F1-score: Harmonic mean of precision and recall, balancing both.

B. Model Performance

The models' evaluation was made on a stratified train-test splits to guarantee that all attack types are equally represented between the training and testing sets.

- **Model 1** achieved an accuracy of 0.86 in binary classification with Decision Tree , demonstrating robust attack detection.

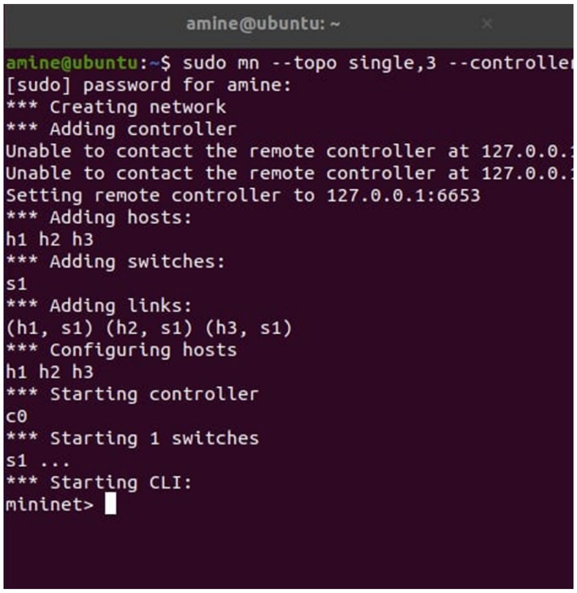
Model	Accuracy	Precision (0)	Recall (0)
SVM (linear)	0.7695	0.95	0.63
SVM (poly)	0.7943	0.96	0.66
SVM (rbf)	0.7909	0.97	0.65
SVM (sigmoid)	0.7202	0.86	0.60
Logistic Regression	0.7623	0.91	0.65
K-Nearest Neighbors	0.7822	0.97	0.64
Random Forest	0.7807	0.88	0.71
Decision Tree	0.8687	0.87	0.90
Gradient Boosting	0.7976	0.83	0.81
XGBoost	0.7972	0.84	0.80
LightGBM	0.8180	0.84	0.84

TABLE I
Model Performance Comparison

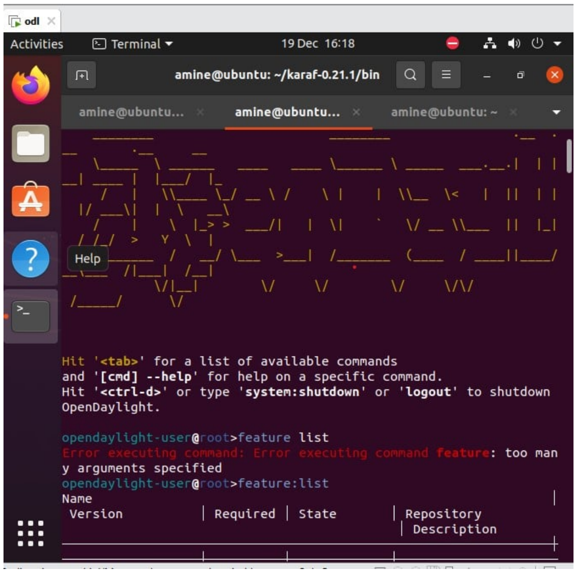
- **Model 2** excelled in multiclass classification, with an accuracy of 0.99, showcasing how effective it is in identifying specific attack types.

Model	Average Accuracy
Decision Tree	0.9997
Random Forest	0.9998
Extra Trees	0.9998

TABLE II
Average Accuracy for Different Models



```
amine@ubuntu:~$ sudo mn --topo single,3 --controlle
[sudo] password for amine:
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.
Unable to contact the remote controller at 127.0.0.
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```



```
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown
OpenDaylight.

opendaylight-user@root>feature list
Error executing command: Error executing command feature: too man
y arguments specified
opendaylight-user@root>feature:list
Name                                                            |
 Version               | Required | State        | Repository
                                                  | Description
```

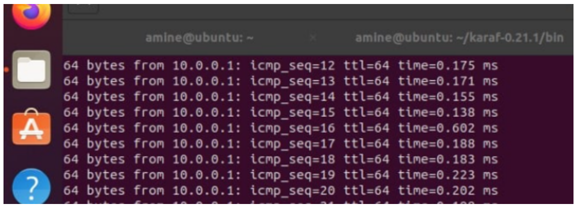
C. Real-Time Simulation Framework

One of the very unique aspects of this research is that it incorporates real-time attack simulation of an environment in the framework by generating dynamic traffic scenarios, testing IDS responsiveness and adaptability under evolving conditions. This approach bridges the gap between theoretical models and real-world deployments.

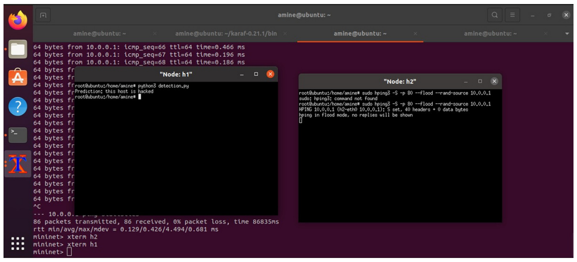
This research simulates an SDN network using Mininet and the OpenDaylight controller to evaluate the proposed IDS. The simulation begins with normal traffic generated from host h2 to host h1, establishing a baseline for typical network behavior. Following this, a DDoS attack is simulated from h2 to h1. The IDS successfully detects the attack, showcasing its ability to identify malicious activity in real-time. This approach bridges the gap between theoretical models and practical implementations by testing the IDS under realistic and evolving network conditions.

The simulation framework allowed for stress-testing of the system under heavy traffic loads, providinginsights into scalability and latency.

Results showed that the proposed IDS still had high detection rates even under the most adverse conditions, thus showing its robustness.



```
64 bytes from 10.0.0.1: icmp_seq=12 ttl=64 time=0.175 ms
64 bytes from 10.0.0.1: icmp_seq=13 ttl=64 time=0.171 ms
64 bytes from 10.0.0.1: icmp_seq=14 ttl=64 time=0.155 ms
64 bytes from 10.0.0.1: icmp_seq=15 ttl=64 time=0.138 ms
64 bytes from 10.0.0.1: icmp_seq=16 ttl=64 time=0.602 ms
64 bytes from 10.0.0.1: icmp_seq=17 ttl=64 time=0.188 ms
64 bytes from 10.0.0.1: icmp_seq=18 ttl=64 time=0.183 ms
64 bytes from 10.0.0.1: icmp_seq=19 ttl=64 time=0.223 ms
64 bytes from 10.0.0.1: icmp_seq=20 ttl=64 time=0.202 ms
```



```
64 bytes from 10.0.0.1: icmp_seq=66 ttl=64 time=0.466 ms
64 bytes from 10.0.0.1: icmp_seq=67 ttl=64 time=0.196 ms
64 bytes from 10.0.0.1: icmp_seq=68 ttl=64 time=0.186 ms
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
64 bytes fr
^C
--- 10.0.0.
86 packets transmitted, 86 received, 0% packet loss, time 86835ms
rtt min/avg/max/mdev = 0.129/0.426/4.494/0.681 ms
mininet> xterm h2
mininet> xterm h1
mininet> 
```

```
root@ubuntu:/home/amine# python3 detection.py
Prediction: this host is hacked
root@ubuntu:/home/amine# 
```

```
root@ubuntu:/home/amine# sudo hping3 -S -p 80 --flood --rand-source 10.0.0.1
sudo: hping3: command not found
root@ubuntu:/home/amine# sudo hping3 -S -p 80 --flood --rand-source 10.0.0.1
HPING 10.0.0.1 (h2-eth0 10.0.0.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

VI. Conclusion

Conclusively, this research introduces for the first time a new approach in intrusion detection for SDN, which integrates machine learning techniques together with a real-time attack simulation framework. This new kind of simulation can dynamically test the intrusion detection system under various threat scenarios, providing an in-depth, real-world-like performance assessment.

Our ML-based approach, using advanced feature selection and dimension reduction techniques and two ML models trained and evaluated on different datasets, showed very good accuracy and adaptability to become a potential way to enhance SDN security.

This work highlights the importance of traditional machine learning methodologies combined with real-time testing environments as a significant step towards practical and proactive network security solutions for systems testing and refinement.

References

[1] G. Logeswari, S. Bose and T. Anitha, “An Intrusion Detection System for SDN Using Machine Learning” Intelligent Automation and Soft Computing Article, Anna University, Chennai, Tamilnadu, India, 16 February 2022

[2] Md. Alamin Talukder, Md. Manowarul Islam, Md Ashraf Uddin, Khondokar Fida Hasan, Selina Sharmin, on ” Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction”, Talukder et al. Journal of Big Data (2024).