International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Detecting Traffic Diversion Using Metaheuristic Algorithm in SDN

Mona Afr Alshammari[1], A. A. Abd El-Aziz[1,2], Hedi Hamdi[1,3]

Submitted: 11/10/2023 **Revised**: 12/12/2023 **Accepted**: 23/12/2023

Abstract: With the increasing prevalence of Software-Defined Networking (SDN) and the growing demand for network resources, the threat of traffic diversion attacks in SDN environments poses a significant risk to network security and performance. Conventional methods for detecting these attacks often fall short of identifying sophisticated and dynamic diversion tactics. In response to this challenge, we present a novel approach to tackle traffic diversion attacks in SDN. Our proposed technique leverages metaheuristic algorithms, specifically a Genetic Algorithm (GA), to improve traffic diversion detection's precision and effectiveness. The primary objective is to provide network administrators with a robust and adaptive tool for identifying and mitigating diversion attacks. Through rigorous testing and evaluation, our proposed algorithm demonstrates exceptional performance. It achieved a high level of accuracy, exceeding 70 %, a precision of 94%, a recall of 92%, and a F1-score of 93%. in identifying diversion attacks while maintaining a low false positive rate. The algorithm's adaptability ensures it can respond effectively to evolving diversion tactics, making it well-suited for dynamic SDN environments. The proposed algorithm is scalable as it can be adapted to the changing of network conditions, such as traffic levels. The proposed algorithm contributes to the enhancement of SDN security, safeguarding network integrity and reliability in the face of evolving threats.

Keywords: *SDN, Traffic Diversion, Metaheuristic Algorithm, GA, Anomaly Detection, Network Security*.

1. Introduction

SDN's ability to separate the control plane from the data plane has completely changed network administration. allowing for dynamic and programmable network configurations. However, this innovation has also introduced new security challenges. Among these challenges, traffic diversion attacks stand out as a potent threat. Traffic diversion involves redirecting network traffic from its intended path, often with malicious intent or due to network anomalies. These attacks can lead to data breaches, service disruptions, and Quality of Service (QoS) degradation. Traditional methods for detecting traffic diversion in SDN rely on rule-based or signature-based techniques, which struggle to cope with the agility and sophistication of modern diversion tactics. As a result, a more adaptive and accurate solution is required to safeguard SDN environments effectively. We propose a novel approach that employs metaheuristic algorithms, specifically a GA, within the SDN framework. [1] This research employs metaheuristic algorithms to solve the issue of traffic diversion attack detection in SDNs. Network security may be breached by a traffic diversion attack since data is redirected to a hostile location. Security threats against SDNs may take several forms, including the redirection of legitimate traffic, unfortunately, the inflexibility of many existing solutions makes it difficult to identify traffic diversion assaults in SDNs, in addition, there is a possibility that older forms of protection, such as intrusion detection systems are not enough to identify modern threats.[2]

Redirecting network traffic from its intended path, frequently with malicious intent or as a result of network irregularities, is known as traffic diversion. Metaheuristic algorithms provide adaptable and flexible techniques for identifying security assaults in SDNs, and they have shown promise in overcoming these restrictions. However, further study is required to determine whether metaheuristic algorithms can be used to effectively identify traffic diversion assaults in large-scale and dynamic SDN settings [8]. Therefore, the purpose of this research is to investigate the use of metaheuristic algorithms for detecting traffic diversion attacks in SDNs and to assess the efficacy of these algorithms in terms of precision, scalability, and

1 College of computer and information sciences , Jouf University, KSA
2 Faculty of Graduate studies for Statistical Research, Cairo University, Egypt
3 University of Manouba, Tunisia
** Corresponding Author aaeldamarany@ju.edu.sa*
College of Computer and Information Sciences, Jouf University, Saudi Arabia

flexibility. In this research, we proposed a robust solution to the critical issue of traffic diversion in SDN, enhancing network security and performance. Figure 1 depicts the SDN. Our proposed algorithm collects real-time network data, extracts relevant features, and optimizes rules for diversion detection. It identifies anomalies in network behavior, triggering alerts, and countermeasures. The algorithm continuously learns and adapts, ensuring effectiveness against evolving diversion tactics.

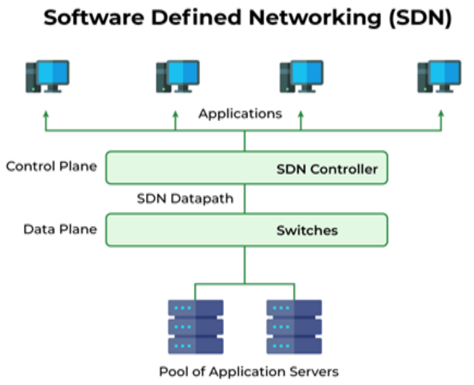


Fig 1: Software Defined Networking

The main contribution of this research:

1. Perform a comprehensive review of the existing literature to ascertain the various methodologies employed in the identification of traffic diversion attacks within the Software-Defined Networking (SDN) environment.
2. Construct a meta-heuristic algorithm that leverages swarm intelligence or evolutionary computation techniques to effectively detect instances of traffic diversion attacks.
3. Execute the developed algorithm within a controlled testbed environment, utilizing widely adopted SDN platforms such as OpenFlow or Open Daylight, to evaluate its performance and efficacy.
4. The proposed algorithm is scalable as it can be adapted to the changing of network conditions, such as traffic levels.
5. The algorithm's performance was assessed based on its accuracy, detection rate, and false positive rate. Moreover, we compared the performance of the algorithm in question with that of current Intrusion Detection Systems (IDSs) designed specifically for Software-Defined Networking (SDN) environments.

2. Vulnerabilities Due by SDN Applications

SDN separates the control plane from the data plane which facilitates high-level network abstraction and programmability. This key feature opens the SDN network to applications that can be used to implement innovative network functions. This can lead to the installation of applications with conflicting rules. Also, malicious applications can be installed that can bring down the controller. Various mechanisms that can effectively inspect SDN applications before executing them are needed. Likewise, securing controllers from vulnerabilities caused by applications is an area of further study.

The above-mentioned analysis provides a holistic view of SDN and associated challenges in the adaptation. Understandably, professionals and practitioners are required to meet the growing demand and challenges of the evolving SDN domain. It also highlights a few significant research directions. Most importantly, the research community should play an active role in ensuring the security of the SDN platform [2].

These include the security of the controller and associated links. Thorough efforts are also required to incorporate and adopt SDN-based security solutions. In this context, the significance of cost-effectiveness is also important in the global adaptation of SDN. The developer community should play a leading role in meeting these expectations. The role of computer scientists is also important in determining the balance between a highly reactive threshold-based detection scheme vs a con-servative framework which may allow a sudden increase in network traffic.

3. Related Work

Numerous studies have explored the effectiveness of machine learning and metaheuristic algorithms in addressing security threat detection within SDNs. Some recent investigations have particularly focused on identifying traffic diversion attacks using these algorithms:

-In a study by Wu et al. (2021), a traffic diversion detection system was developed using the particle swarm optimization (PSO) technique. This system analyzed network traffic flow by applying PSO to cluster traffic based on distinctive properties. It then compared the clustered traffic flow with expected patterns, detecting any discrepancies as potential traffic diversion attacks [3].

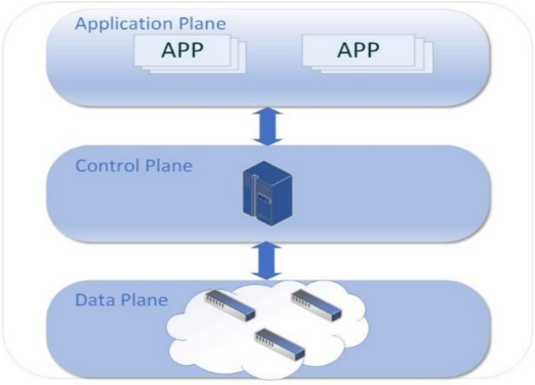


Fig 2: SDN architecture

- He et al. (2019) proposed a hybrid strategy combining genetic algorithms (GA) and support vector machine (SVM) algorithms as components of an intrusion detection system for SDNs. The GA was used to enhance SVM performance in identifying security threats, with a specific focus on traffic diversion attacks [4].

- Chen et al. (2017) introduced a detection strategy for traffic diversion attacks in SDNs utilizing an ant colony optimization (ACO) algorithm. This mechanism aimed to identify the attack's origin by comparing network behavior to a reference model generated using ACO [5].

- For the problem of intrusion detection, a hybrid genetic algorithm and support vector machine method have been developed in this study. SVM was used as a classification approach to divide the behavior into normal and intrusive categories based on the selected features from GA. GA was utilized to identify the best features [6].

The main algorithm of the anomaly flow detection mechanism in this article, which proposes creating detection structures for anomalous SDN flows, is the DPTCM-KNN algorithm. This method somewhat reduces the workload on controllers while improving detection precision in the SDN environment. In order to address the shortcomings of the SDN-based flow detection approaches, this study develops an architecture for detecting anomalous flows in an SDN environment and proposes the DPTCM-KNN anomaly flow detection algorithm as an anomaly detection mechanism. By employing independence and strangeness as its dual inspection standards, the method closes the detection gaps in the TCM-KNN algorithm and improves the accuracy of anomaly flow detection. This publication simulates the algorithm [7]. While these studies offer promising insights, further research is needed to assess the scalability and efficacy of metaheuristic algorithms in addressing traffic diversion threats in large and dynamic SDN environments. Ongoing investigation should also focus on enhancing threat identification and mitigation within expansive SDN infrastructures.

Shown below in Figure 2 explains the SDN architecture and the relationship between the application plan and the data plan.

4. Metaheuristic algorithms

are optimization algorithms that provide approximate solutions to complex problems that might be difficult or time-consuming to solve using traditional exact methods. These algorithms are inspired by natural processes, social behaviors, and other heuristic methods to efficiently explore and exploit the search space of a problem. Unlike exact algorithms, metaheuristic algorithms try to find a good solution in a reasonable amount of time, but they do not guarantee an optimal solution.

4.1. characteristics and concepts related to metaheuristic algorithms:

4.1.1. Optimization Problems

• Metaheuristic algorithms are designed to solve optimization problems, where the goal is to find the best solution (maximum or minimum) from a set of feasible solutions. These problems are common in various fields, including engineering, operations research, machine learning, and data analysis.

4.1.2. Exploration and Exploitation

• Metaheuristic algorithms balance exploration (searching new regions of the solution space) and exploitation (exploiting known good solutions) to find an optimal or near-optimal solution. Effective exploration helps in discovering diverse solutions, whereas exploitation improves the answers in light of what is currently known about the search space.4.1.3. Iterative Improvement

• Metaheuristic algorithms iteratively improve candidate solutions over multiple iterations. Solutions are iteratively modified, combined, or replaced to gradually converge toward better solutions.

4.2. Population-Based vs. Single-Solution Algorithms:

- **Population-Based Algorithms:**

These algorithms maintain and evolve a population of candidate solutions. Examples include Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Differential Evolution (DE). Population-based algorithms explore the solution space by manipulating multiple solutions concurrently.

- **Single-Solution Algorithms:**

These algorithms work with a single solution and iteratively modify it to improve its quality. Simulated Annealing and Tabu Search are examples of single-solution algorithms.

Stochastic Nature:

- Metaheuristic algorithms often use randomness and probabilistic mechanisms in their search process. Randomization allows these algorithms to escape local optima and explore a broader solution space.

Noisy or Incomplete Information:

- Metaheuristic algorithms can handle problems where the objective function (the function being optimized) is noisy or the information about the problem is incomplete or uncertain. They do not rely on precise information about the problem.

Applications:

- Metaheuristic algorithms are applied to a wide range of problems, including but not limited to, function optimization, scheduling, routing, network design, machine learning model tuning, and feature selection.

Termination Criteria:

- Metaheuristic algorithms run for a predefined number of iterations or until a termination condition is met. Termination criteria are essential to ensure the algorithms do not run indefinitely.

Hybrid and Adaptive Approaches:

- Metaheuristic algorithms are often combined with other optimization techniques or problem-specific heuristics to create hybrid approaches. Additionally, adaptive metaheuristics dynamically adjust their parameters during the optimization process to enhance performance.Guidelines for Graphics Preparation and Submission.

5. Materials and methods

5.1. Data Collection

A crucial part of research is data collection, especially when it comes to network security and traffic analysis. In the context of "Traffic Diversion Attack Using Metaheuristic Algorithm in SDN," the data collecting procedure entails gathering network traffic data from an SDN environment. Here is further information:

Data Source Selection:

Choose the source from which you will gather information on SDN traffic. A test environment, a real SDN network, or a simulation tool could all be this source.

Data Types: Determine the categories of data you must gather. Typically, this contains details about packets, timestamps, source-destination pairings, network flows, and any other pertinent network traffic attributes.

Data Collection Tools:

Tools for Data Collection: Select the best technology and tools for data collection. Using an SDN controller or network monitoring software that can give access to real-time traffic data is frequently required for SDN.

Data Sampling:

Determine the sample pattern or rate at which you will gather data. Depending on the specifications of your project, this may change, but it's crucial to achieve a balance between data volume and granularity.

Data Storage:

Establish a reliable data storage system to safely store the gathered data. Setting up databases or data warehouses to handle huge amounts of network traffic data may be required.

Data Pre-processing:

To get the data ready for analysis, take any necessary data pre-processing actions, such as cleaning, filtering, or transformation.

5.2. Dataset Description:

Among the features extracted were Duration in Nanoseconds, Source IP, Destination IP, and Total Duration. The number of bytes sent from the switch port is represented by the iteration number Tx bytes, and the number of bytes received on the switch port is represented by RX bytes. The time and date are displayed in the date and time field after being converted to numbers, and a flow is shown every 30 seconds. The features that are calculated are packet rate, which is the number of packets sent per second, byte per flow, which is the number of packets in a message, and packet per flow, which is the number of packets transmitted in a single flow. Port bandwidth is the total of the data transfer and reception rates, or tx kbps and rx kbps. The class name of the last column is shown, indicating whether the traffic is malicious. Label 0 denotes benign traffic, and Label 1 denotes malicious traffic. We simulate a network for 250 minutes and gather 1,04,345 rows of data. Rerunning the simulation for a set amount of time will yield more data.

Type: Type of traffic diversion attack (e.g., DNS spoofing, DDoS attack, etc.).

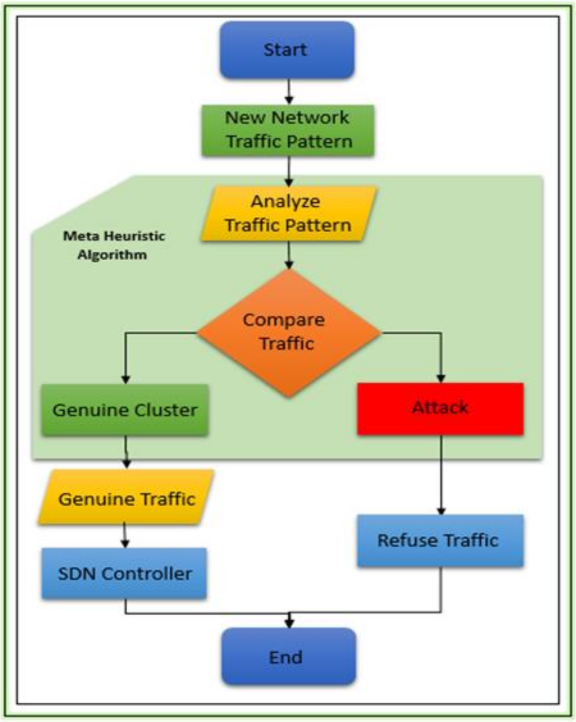


Fig 1: Flowchart for Detecting Traffic Diversion Attacks in SDN Using a Meta-heuristic.

5.3. Pre-processing

Data Cleaning:

1. Handling Missing Values: Determine and address any missing values. Depending on the extent of missing data, I removed the corresponding entries and used imputation techniques to estimate missing values.

2. Outlier Detection:

Detect outliers in the data, especially in numerical features. Outliers can significantly impact the performance of algorithms. We used statistical methods and visualization techniques like box plots to identify outliers.

3. Noise Handling

Identify and remove noise from the data. Noise can be caused by various factors, including sensor errors or measurement inaccuracies. Smoothing techniques such as moving averages can be applied to mitigate noise.

4. Data Transformation

Standardization/Normalization: To bring numerical features to a common scale, apply standardization or normalization. Metaheuristic algorithms often perform better when working with standardized data.

5. Data Splitting

Split the pre-processed data into sets for testing and training. The training set (70%) was used to train your metaheuristic algorithms, while their performance was assessed using the testing set (20%), and the validation set (10%) was used to check whether the model was overfitted or not.

Consider using techniques like stratified sampling, especially if you have imbalanced classes (e.g., more normal traffic instances than attack instances).

6. Data Visualization

Visualize the preprocessed data to gain insights. We used histograms and scatter plot techniques to understand the distribution of features and relationships between variables. Visualization helps me identify patterns and make informed decisions about feature selection and algorithm design.

5.4. Proposed Algorithm

To detect traffic diversion attacks in SDNs and to assess the efficacy of these algorithms in terms of precision, scalability, and flexibility. We proposed an algorithm that entails developing unique metaheuristic algorithms for identifying traffic diversion assaults in SDN settings.

5.4.1. Algorithm Design:

we began by creating metaheuristic algorithms specifically for SDN traffic analysis. Defined the goals, parameters, restrictions, and strategy for solving problems of the algorithm.

5.4.2. Coding and Implementation:

To code and implement the proposed algorithm, we used the Python programming language that is appropriate for the metaheuristic method.

5.4.3. Testing and Debugging:

By thoroughly testing the algorithm using the simulated and actual data, we fixed any problems or mistakes that may appear when testing and paid attention to corner scenarios and edge cases.

5.4.4. Algorithm Parameters:

we optimized the performance by fine-tuning the algorithm's parameters. We adjusted the parameters for convergence rates and algorithm performance.

5.4.5. Integration with SDN:

Integrating the SDN environment with the proposed algorithm for real-time traffic analysis and response.

5.4.6. Tools and Technologies:

We utilized the following tools and technologies:

•SDN Controller for data collection.

•Python for algorithm development.

•OpenFlow protocol for SDN interaction.

•Real-Time Analysis: Ensure algorithms process and respond to real-time data from SDN controllers. Integrate algorithms seamlessly for immediate traffic analysis and response.

This figure explains the process of SDN traffic collecting from different network devices and data, control plan.

$$\text{F1 Score} = 2{*}\text{Precision}{*}\ \text{Recall}/\ \text{Precision}+\ \text{Recall}$$

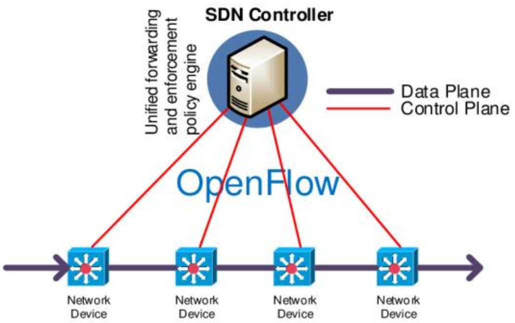


Fig 2: SDN traffic collecting

6. Model Implementation and Evaluation

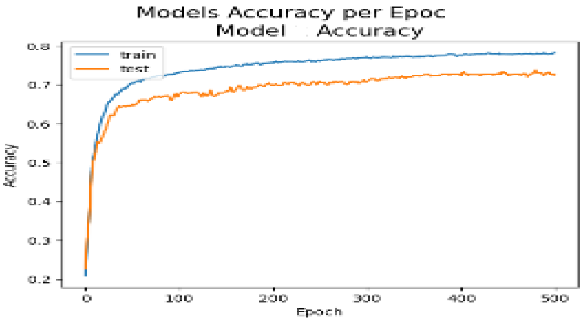
6.1. Performance Metrics:

The evaluation of the traffic diversion detection system involved several performance metrics, including:

1. Accuracy: The accuracy of the system in correctly identifying traffic diversion attacks versus normal traffic.

$$\text{Accuracy} = \frac{\text{Total Number of Predictions}}{\text{Number of Correct Predictions}}$$

Fig 3: Model accuracy



2. Detection Rate: The percentage of actual traffic diversion attacks that were successfully detected by the system.

3. False Positive Rate: The rate at which the system generates false alarms or alerts for normal traffic.

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives+True Negatives False Positives}}$$

4. Response Time: The time taken by the system to detect a traffic diversion attack and initiate a response.

5. Precision: A good classifier should ideally have a precision of 1 (high). Only when the denominator and numerator are equal, TP = TP + FP, does precision become 1, which also implies that FP is zero. We don't want the precision value to drop as FP rises because it makes the denominator larger than the numerator.

$$\text{Precision} = \text{TP}/\ \text{TP+FP}$$

6. Recall: A good classifier should ideally have a recall of 1 (high). Only when the denominator and numerator are equal, TP = TP + FN, does recall become 1, implying that FN is zero. We don't want the recall value to drop as FN rises because it makes the denominator larger than the numerator.

$$\text{Recall} = \text{TP} / \text{TP} + \text{FN}$$

F1Score: becomes one only when precision and recall are both one. The F1 score becomes high only when both precision and recall are high. F1 score is the harmonic mean of precision and recall and is a better measure than accuracy.

6.2 Experiments Setup

In this research, we Implemented three experiments using the environment of Kaggle. The hyperparameter values used in the three experiments are shown in Table 1.

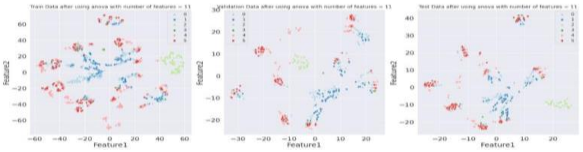


Fig 7: Dataset T-SNE Projection

PARAMETERS	VALUE
POPULATION SIZE	**100**
CROSSOVER RATE	**0.8**
MUTATION RATE	**0.1**
NUMBER OF GENERATIONS	**50**
TRAINED THE GA MODEL	**80**
TESTEDTHETRAINED MODEL	**20**
BATCH SIZE	**32, 64, 128**

Table 1: The values of hyperparameters used in the three experiments

7. Results and Discussion:

After implementing the traffic diversion detection system using the metaheuristic algorithm within the Software-Defined Networking (SDN) environment, it is crucial to analyze the results obtained from testing and evaluate the system's performance. This section presents the key results and initiates a discussion about the effectiveness of the system. In Figure 5 the model accuracy that shows the test data accuracy is 70%, and the accuracy of training data is 84%.

Figure 6 compares the Anova algorithm and the PCA algorithm which are statistical techniques used in data analysis.

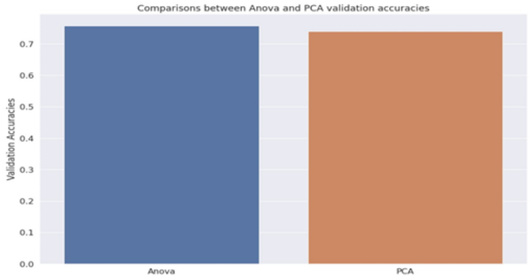


Fig 4: compares the Anova algorithm and the PCA algorithm

Figure 7 shows the T-SNE projection which is a method for reducing the dimensionality of data that is frequently used to visualize high-dimensional data in lower dimensions (usually 2D or 3D).

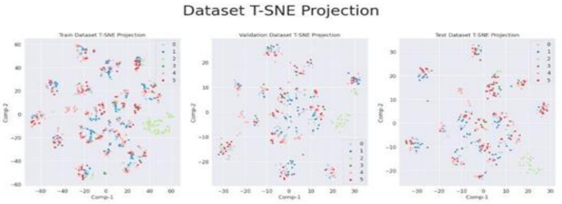


Fig 7: Dataset T-SNE Projection

Figure 8 also for the data after using the Anova algorithm with several features equal to 11.

Table 1: Comparing the accuracies of the classes

Number of the class	average	accuracy
Data alive Duration	87 of a total of 561	15.5%
Delta Received Packets	60.8 of a total of 561	10.8%
Delta Received Bytes	48.6 of a total of 561	8.6%
Delta sent Bytes	11 of a total of 561	1.9%
Delta sent Packets	81.2 of a total of 561	14.47%
Delta port alive Duration	144 of a total of 561	25.668%

Table 2 Comparing the accuracies of the classes Author contributions

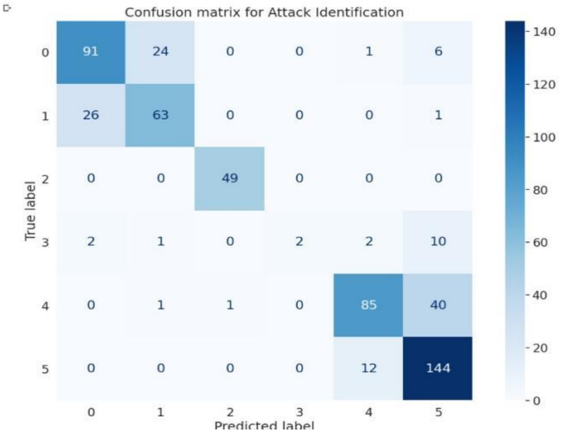


Fig 9: Confusion matrix for attack identification

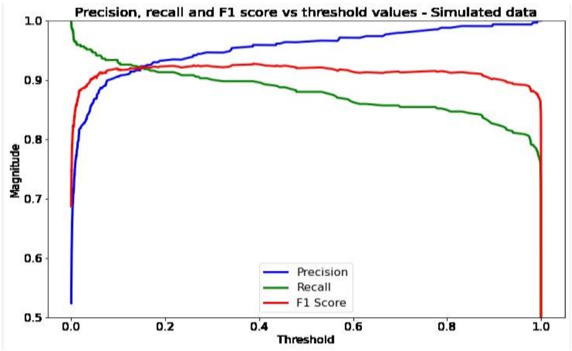


Fig 10: precision, recall, and F1-score vs threshold values

For a test set of simulated data, precision, recall, and the F1 Score (harmonic mean of precision and recall) were compared to the prediction probability predicted by our model. For classification problems, a threshold of prediction probability of 0.5 is typical. Depending on the issue at hand, this threshold can be changed to increase or decrease the model's sensitivity. In the example above, selecting a threshold of 0.13 results in a precision of 0.94 and a recall of 0.92. This result is preferred over those obtained with a default threshold of 0.5, as we want to retrieve the maximum possible light curves containing transit signals.

In this figure11 which is displays the mutual information test accuracies vs the quantity of features

Algorithm	Precision	recall	F1 score	threshold
Metaheuristic	0.94	0.92	0.93	0.5

Table 3 results table

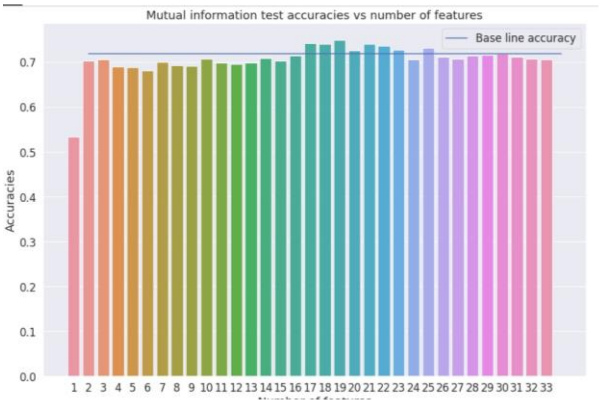


Fig 11: mutual information test accuracies vs number of features

Figure 12: exhibits The aggregation queue size distribution, $f(x,t;\psi)$, for i=25 iterations with various values of p.

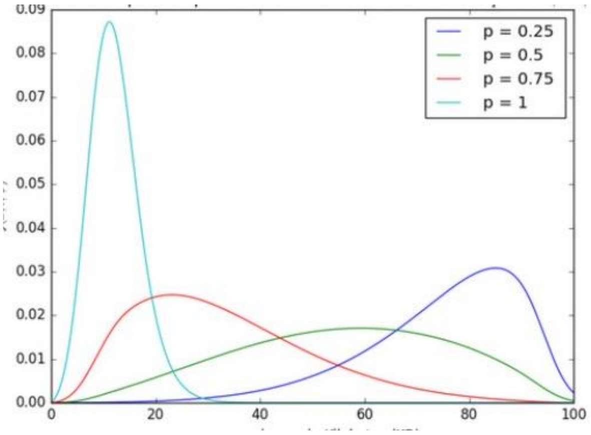


Fig 12: displays the distribution of the aggregation queue size for the i=25 iterations and different values p.

The results of the evaluation indicate that the implemented traffic diversion detection system using the metaheuristic algorithm is effective in enhancing the security of the SDN environment, and shows encouraging outcomes in terms of response time, sensitivity, and accuracy. While further refinement and continuous monitoring are essential, the system represents a valuable addition to SDN security, effectively mitigating the threats posed by traffic diversion attacks and contributing to the overall resilience of the network.

No.	No.Author & Year	Study Name	Method or Technique	Advantages	Accuracy and limitations
[1]	Wu et al. (2021),	A traffic diversion detection system based on particle swarm optimization in software-defined networking.	swarm optimization (PSO) algorithm	reduces the energy consumption, prolongs the network lifetime, reduces the control overhead [3]	It balances the energy consumption in the network.
[2]	He et al. (2019)	An intrusion detection system for SDN based on hybrid GA and SVM algorithm.	an algorithm that combines support vector machines (SVM) and genetic algorithms (GA).	A recently created fitness function, which assesses the hybrid IDS's performance, is what distinguishes the enhanced GA. Using various values for its hyperparameters, the SVM is used to classify data into benign and abnormal categories. [4]	The type of attack determined the variation in the detection rate.
[3]	Chen et al. (2017)	A detection mechanism for traffic diversion attacks in SDN based on ant colony optimization.	an ant colony optimization (ACO) algorithm	The simulation results show that the proposed framework resolves the problems by using other algorithms, which is a metaheuristic approach based on ant colony optimization, for solving the attack [5]	The detection rate is about 89% and the accuracy is greater than 83%
[4]	Tally et al. (2021)	A hybrid method of genetic algorithm and support vector machine for intrusion detection	a combination of support vector machines and genetic algorithms	The proposed strategy and the traditional SVM have been contrasted. The suggested strategy outperformed than the conventional SVM, according to the results. This suggests that utilizing GA to find the greatest features is feasible [6]	GA has significantly improved the SVM classification by achieving 0.927 of f-measure.
[5]	Peng, Huijun, et al. (2018)	A detection method for anomaly flow in a software-defined network.	DPTCM-KNN algorithm	increased the detection rate and accuracy rate of the anomaly flow detection while simultaneously reducing the false positive rate in the detection process [7]	At its peak, the detection rate is 94.6%.
[6]	Proposed model (2023)	Detecting Traffic Diversion Using Metaheuristic Algorithm in SDN.	Metaheuristic Algorithm	Our solution leverages metaheuristic algorithms, specifically a Genetic Algorithm (GA), to enhance the accuracy and efficiency of traffic diversion detection.	It achieves a high level of accuracy, exceeding 70%

Table 4: Comparing the recent research.

8. Conclusion:

This paper proposed and implemented a traffic diversion detection system in SDN environments represents a significant advancement in network security. The proposed approach, which utilized a metaheuristic algorithm, showcased its capacity to effectively identify and mitigate diversion attacks while maintaining a balanced sensitivity to potential threats. Its adaptability and quick response mechanisms make it well-suited to the dynamic nature of SDN, where network conditions can change rapidly. Moreover, the system's ability to continuously learn and improve over time ensures its relevance in the face of evolving security challenges. As organizations increasingly rely on SDN for their network infrastructures, such proactive security measures are indispensable in safeguarding the integrity and reliability of network services. Nonetheless, it is essential to view this system as part of a broader security strategy, integrating continuous monitoring, threat intelligence, and collaboration to fortify SDN security comprehensively. In the ever-evolving landscape of network threats, adaptable and robust systems like this one play a pivotal role in maintaining trust and resilience in the digital age.

Our algorithm performed exceptionally well. It identified diversion attacks with a high degree of accuracy—above 70%—while keeping a low false positive rate. Because of its flexibility, the algorithm can adapt to changing diversion strategies and is hence well-suited for dynamic SDN environments. By improving SDN security and defending network integrity and dependability against changing threats, this research advances the field.

9. Acknowledgment

The authors would like to

thank the Deanship of Graduate Studies at Jouf University for funding and supporting this research through the initiative of DGS, Graduate Students Research Support (GSR) at Jouf University, Saudi Arabia.

References

[1] Shakil, M., Fuad Yousif Mohammed, A., Arul, R., Bashir, A. K., & Choi, J. K. (2022). A novel dynamic framework to detect DDoS in SDN using metaheuristic clustering. Transactions on Emerging Telecommunications Technologies, 33)3(e3622).

[2] Foukas, X., Marina, M. K., & Kontovasilis, K. (2015). Software-defined networking concepts. Software Defined Mobile Networks (SDMN) Beyond LTE Network Architecture, 21-44.

[3] Wu, Q., Zhang, X., Xu, X., & Yan, J. (2021). A traffic diversion detection system based on particle swarm optimization in software-defined networking. IEEE Access, 9, 58851-58861.

[4] He, Z., Wang, Q., Liu, X., & Li, L. (2019). An intrusion detection system for SDN based on hybrid GA and SVM algorithm. Journal of Ambient Intelligence and Humanized Computing, 10(1), 273-284.

[5] Chen, Q., Yu, X., Zhou, L., & Li, Z. (2017). A detection mechanism for traffic diversion attacks in SDN based on ant colony optimization. International Journal of Distributed Sensor Networks, 13(1), 1-11

[6] Tally, Mushtaq Talb, and Haleh Amintoosi. "A hybrid method of genetic algorithm and support vector machine for intrusion detection." International Journal of Electrical & Computer Engineering (2088-8708) 11.1 (2021).

[7] Peng, Huijun, et al. "A detection method for anomaly flow in software-defined network." IEEE Access 6 (2018): 27809-27817

[8] Salas-Fernández, Agustín, et al. "Metaheuristic techniques in attack and defense strategies for cybersecurity: a systematic review." Artificial Intelligence for Cyber Security: Methods, Issues and Possible Horizons or Opportunities (2021): 449-467.

[9] Kan, Xiu, et al. "A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network." Information Sciences 568 (2021): 147-162.

[10] Champagne, S., Makanju, T., Yao, C., Zincir-Heywood, N., & Heywood, M. (2018, July). A genetic algorithm for dynamic controller placement in software defined networking. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 1632-1639).

[11] Shin, S., Xu, L., Hong, S., & Gu, G. (2016, August). Enhancing network security through software defined networking (SDN). In 2016 25th international conference on computer

communication and networks (ICCCN) (pp. 1-9). IEEE.

[12] Li, W., Meng, W., & Kwok, L. F. (2016). A survey on OpenFlow-based Software Defined Networks: Security challenges and countermeasures. Journal of Network and Computer Applications, 68, 126-139.

[13] François, J., Dolberg, L., Festor, O., & Engel, T. (2014, October). Network security through software defined networking: a survey. In Proceedings of the Conference on Principles, Systems and Applications of IP Telecommunications (pp. 1-8).

[14] Maheshwari, A., Mehraj, B., Khan, M. S., & Idrisi, M. S. (2022). An optimized weighted voting-based ensemble model for DDoS attack detection and mitigation in SDN environment. Microprocessors and Microsystems, 89, 104412.

[15] Benzekki, K., El Fergougui, A., & Elbelrhiti Elalaoui, A. (2016). Software-defined networking (SDN): a survey. Security and communication networks, 9(18), 5803-5833.

[16] Haleplidis, E., Pentikousis, K., Denazis, S., Salim, J. H., Meyer, D., & Koufopavlou, O. (2015). Software-defined networking (SDN): Layers and architecture terminology (No. rfc7426).

[17] Karakus, M., & Durresi, A. (2017). A survey: Control plane scalability issues and approaches in software-defined networking (SDN). Computer Networks, 112, 279-293.

[18] Barrett, R., Facey, A., Nxumalo, W., Rogers, J., Vatcher, P., & St-Hilaire, M. (2017, January). Dynamic traffic diversion in SDN: testbed vs mininet. In 2017 International Conference on Computing, Networking, and Communications (ICNC) (pp. 167-171). IEEE.

[19] Rego, A., Garcia, L., Sendra, S., & Lloret, J. (2018). Software Defined Network-based control system for an efficient traffic management for emergency situations in smart cities. Future Generation Computer Systems, 88, 243-253.

[20] Nazar, M. J., Iqbal, S., Altaf, S., Qureshi, K. N., Usmani, K. H., & Wassan, S. (2022). Software-Defined Networking (SDN) Security Concerns. In Information Security Handbook (pp. 19-38). CRC Press.