Design and Implementation of Post-Detection of Denial of Service (DoS) as a Mitigation System (PDDMS) Based on Dynamic Access Control List Algorithm

by Adian Fatchur Rochim

Submission date: 28-Jul-2022 11:12AM (UTC+0700)
Submission ID: 1876065343
File name: 1570766830.pdf (597.64K)
Word count: 3947
Character count: 20906

Design and Implementation of Post-Detection of Denial of Service (DoS) as a Mitigation System (PDDMS) Based on Dynamic Access Control List Algorithm

Adian Fatchur Rochim
Department of Computer Engineering, Faculty of Engineering
Universitas Diponegoro
Semarang 50275, Indonesia
adian@ieee.org

Fahmi Maghrizal Mochtar
Department of Computer Engineering, Faculty of Engineering
Universitas Diponegoro
Semarang 50275, Indonesia
fmmochtar@students.undip.ac.id

Adnan Fauzi
Department of Computer Engineering, Faculty of Engineering
Universitas Diponegoro
Semarang 50275, Indonesia
adnan@live.undip.ac.id

***Abstract*— Computer networking maintenance and monitoring have been essential things. A human administrator could not monitor the whole resources for 24 hours and take action directly in inactive hours when an incident occurs. Automating the network appliance with the integration of an attack detection system could help solve the problem. This study mainly focuses on mitigating network attacks using the Dynamic Thresholding algorithm as a detection and mitigation system based on network automation using the Dynamic Access Control List algorithm. The data used for this research is self-generated in a virtual environment and a mitigation system written in Python to automate the router configuration through REST API. Prototype of the mitigation system, namely post-detection of DoS as a Mitigation System (PDDMS). The system testing phase results show that the mitigation system has an average of 1.57 seconds response time to configure ACL for one router. The implementation evaluated using Confusion Matrix shows 0% results of True-Positive Rate in the generated dataset, with 23.01% of accuracy and no positive results detected, which resulted in no response taken by mitigation system.**

Keywords*— *Computer Network, Network Automation, DoS Mitigation, Mitigation System, RESTful API

I. INTRODUCTION

Computer networking maintenance and monitoring have been crucial points in terms of resource management. Mainly, an experienced network administrator who has a responsibility for this would take action directly. A human network administrator does not have a 24-hour endurance to monitor and handle a single activity if an action needs to be taken precisely and immediately when a critical event occurs. A mitigation system is required to solve this problem.

Security aspects are also considered when it comes to defending existing resources. The ever-existing attacks in networking need to be mitigated. Cisco stated that cyber-attacks such as Distributed Denial of Service would likely increase to 15.4 million globally [1].

The evolving networking automation technologies lead to network automation, improving management capabilities of the network appliances in an infrastructure. Prior research [2][3] has successfully implemented an automation system for network appliances. However, those research only apply essential network configuration functions, such as interface IP address configuration, routing, backup, and restore configuration features. Moreover, those research does not implement either the security measures, or the security management system to protect the network resources. An automated system that could take a role in preventing or mitigating attacks might address the issue [4].

Aziz et al., in 2019, explained how ELK Stack that consists of Elasticsearch, Logstash, and Kibana used in large-scale infrastructure as a logging monitoring system, which stores data consists of logs of network appliances in the managed network [5].

Rafi et al., in 2020, explained how multiple Cisco CSR1000v routers could be configured and managed by automating it through a RESTful API using an application written in Python as a tool to configure those devices [2]. The application can manage the device configuration through a Django web-based interface, replacing the command line interface-based configuration for quicker usability.

Ramprasath et al., in 2021, stated how ingress filtering works for mitigating DDoS attacks in a Software-defined network by dynamically configuring access control lists in OpenFlow switches [6], which is called by the Dynamic Access Control List algorithm. The system uses ACL policies to mitigate the traffic by generating new rules if the detection system detects any positive DDoS attacks.

Yadav et al., in 2018, explained that the Access Control List configuration could be implemented on Cisco routers as a solution to mitigate DDoS attacks by configuring Access Control Lists in the router to filter connections based on IP addresses so it could prevent attack connections from other networks [7].

David et al., in 2019, explained how the Dynamic Thresholding algorithm could be used to analyze and detect DoS attacks [8]. This method compares the aggregation results of four attributes in the header of each packet entering the network by calculating the moving average and moving variance for each specific time interval. The advantage of this method is the lower consumption of computational resources than other variants of Anomaly-based methods, such as the ARIMA model and chaos theory [9] or the Machine Learning model [10].

This study aims to expand prior research, mainly the network automation system, by building a DoS attack mitigation system by implementing existing methods.

In this paper, we built a prototype of the mitigation system, namely Post-Detection of DoS as a Mitigation System (PDDMS), to mitigate attacks related to Denial-of-Service, which could potentially exhaust existing resources in a particular network. Network automation as a mitigation system was chosen since a router can block specific sources from accessing targeted networks.

The rest of this paper is structured as follows: in the next section, we describe the methodology used to build the system. The third section gives the result and discussion of the system implementation, and the fourth section concludes the paper.

II. RESEARCH METHODOLOGY

The main goal of this research is to design and implement a model for the mitigation of an attack that happened in computer networks. This research expands the past research conducted by A. Rafi in 2020, which already built a system to configure router devices through RESTful API.

The research methodology comprises steps explained as follows:

1. Rebuilding an automation system based on the past research.
   This phase started by implementing a system used in the past research. In this phase, the main reference used is the research conducted by A. Rafi in 2020 about network automation using RESTful API.
2. The implementation of a DDoS attack detection system.
   In this phase, the Dynamic Thresholding algorithm is implemented to detect the attack. A team partner has already conducted this work before.
3. The implementation of attack mitigation system based on Dynamic Access Control List method.
   In this phase, the system is designed according to the system design and topology that has been made based on previous research.
4. System Testing and results phase.
   In this phase, the result of the research can be concluded based on the system test output.

In this study, we are implementing the methodology as shown in Figure 1.

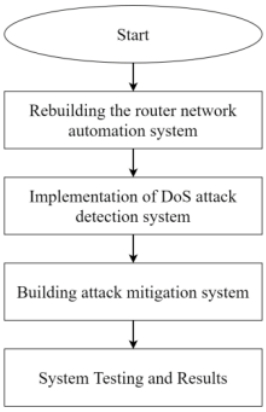

Figure 1 Research methodology flowchart

Figure 2 shows the relation in general about the networking with two systems, namely a DDoS detection system and a mitigation system. This paper only discusses the design and implementation of the mitigation system, which is based on the Dynamic Access Control List method.

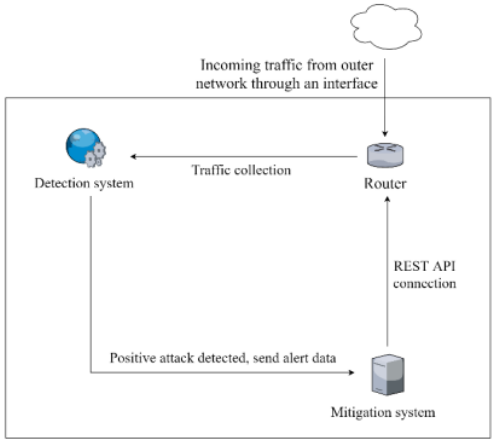

Figure 2 System Design of Mitigation System (PDDMS)

Figure 3 shows the topology used in this experiment, which is also related to the system design in Figure 2.

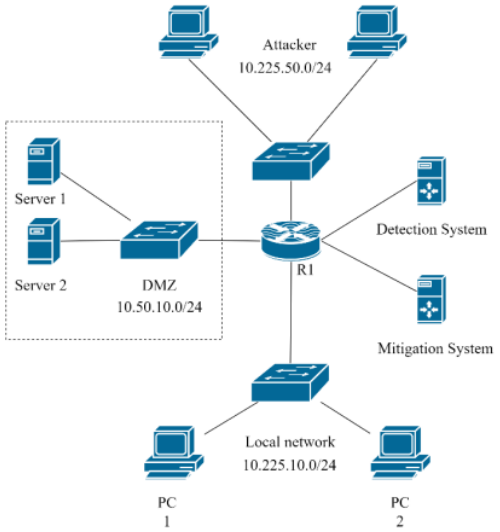

Figure 3 Scenario and implementation of PDDMS testing

Table 1 shows the list of devices implemented in the scenario, as shown in Figure 3.

TABLE I. LIST OF DEVICES IMPLEMENTED IN THE SCENARIO

Device Name	IP Address	Roles
Attacker 1	10.225.50.11	Sends malicious traffic to servers
Attacker 2	10.225.50.12	Sends malicious traffic to servers

TABLE I. (CONTD.)

Device Name	IP Address	Roles
Mitigation system	10.100.10.2	Automate configuration to block attackers
Detection system	10.100.10.151	Detecting attacks
Server 1	10.50.10.11	Targeted server (Port 80 opened)
Server 2	10.50.10.12	Backup of Targeted server (Port 80 opened)
PC 1	10.225.10.11	Connection test
PC 2	10.225.10.12	Connection test

In this research, we implement an algorithm to mitigate the attack in the network using the Dynamic Access Control List, which is an algorithm used in research conducted by Ramprasath et al. in 2020 to mitigate networking attacks. The main idea of the algorithm is to generate rules to filter the traffic based on the IP address of the attacker to the targeted system when the occurring attack has been detected.

The Dynamic Access Control List method used in this experiment is explained as follows:

1. The system will collect the information of the attack based on the alert received from the detection system.
2. The system will inspect the source IP address of the attacker and the destination IP address, and the destination port of the targeted system.
3. After the parameters have been collected, the system will generate the Access Control List rule to block the connection.
4. The generated rule will be sent to the router as an instruction through REST API to block the connection based on the newly generated configuration.

Figure 4 shows the flowchart of the Dynamic Access Control List method used in this research.

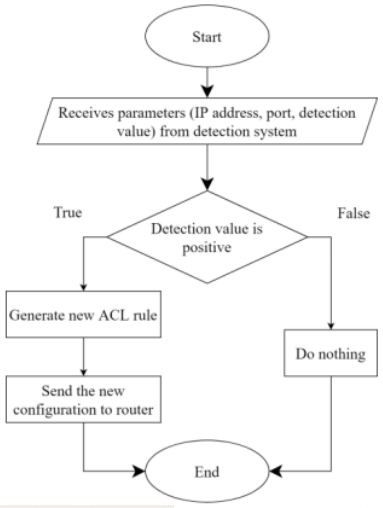

Figure 4 Flowchart of the Dynamic Access Control List algorithm

A. *Rebuilding the system based on past research*

This research reimplements the system designed by A. Rafi as the basis for implementing this mitigation system, which uses RESTful API-based network automation to configure the router. All the tests and simulations were performed on a single computer in a virtual environment in this experiment.

The software used in this experiment are Cisco CSR1000v, Ubuntu deployed as virtual machines, Python 3.8 with Django 3.2 for the application, GNS3 for topology and networking simulation, and QEMU as a hypervisor for virtualizing routers and servers.

The main functional application requirements are automation of the Access Control Lists based on the Dynamic Access Control List algorithm, one of the main functions needed for mitigation integrated with the attack detection system.

B. *Implementation of DDoS attack detection system*

The DoS detection system uses the Dynamic Thresholding Algorithm [8], a DoS attack detection algorithm written by J. David in 2019. This algorithm compares the aggregation results of each packet's four header attributes entering the network, calculated based on the moving average and moving variance for each specific time interval. An attack is considered in progress if the calculation results of the four attributes cross the limit or threshold simultaneously. The algorithm consideration of the decision to classify the attack is based on limitations of the four header attributes of the data flow through the transmission medium at a certain period.

C. *Implementing Attack Mitigation System based on Dynamic Access Control List algorithm*

The next stage is the implementation stage of the mitigation system. The development of the mitigation system uses the Python Requests library, which is used as automation and to communicate with RESTful API. In this stage, the implementation and testing stages were performed and simulated in GNS3.

The mitigation system uses the Dynamic Access Control List [6], an algorithm used to mitigate DoS attacks. This algorithm will automatically generate rules to drop packets based on the source IP address, destination IP address, and the destination port if the result of the detection is classified as a positive attack.

Since the system utilizes Cisco routers, the mitigation system utilizes a feature called Access Control List policies to filter the traffic based on source IP address. The DoS attack mitigation algorithm will apply new ACL rules based on connection details for denial of a packet entry toward the targeted network [7]. REST API is used to automate the router configuration and generate the Access Control List rules for mitigation. The designed mitigation system will directly communicate with the router for mitigation purposes, as shown in Figure 2.

The workflow for implementing the DoS mitigation system, as shown in Figure 2, is as described as follows:

1. Incoming traffic from other networks passes through the CSR1000v router.
2. CSR1000v router performs constant traffic to the detection system

3. The detection system receives the data and will analyze incoming traffic based on the Dynamic Thresholding algorithm.
4. If the detection system detects positive results, it will trigger the mitigation system to block the attacker based on the Dynamic Access Control List algorithm by sending REST API to generate ACL rule to block the attacker connection.

Table 2 shows the REST API endpoint list used to configure the device for attack mitigation in this experiment, which shows the related configuration URL endpoint and the HTTP method.

TABLE II. REST API REQUESTS

Configuration	HTTP Method	Endpoint	Response
Create an Access Control List	POST	https://ip_address:55443/api/v1/acl	201 Created
Modify Access Control List Rules	PUT	https://ip_address:55443/api/v1/acl/{acl-id}	200 OK
Get Access Control Lists Information	GET	https://ip_address:55443/api/v1/acl/{acl-id}	200 OK

The topology consists of one router, two attackers, two virtual servers in DMZ, one server for detection system to analyze incoming traffic passing through the routers, bridged, and directly connected to both routers.

In this experiment, R1 is used for both traffic header collection and mitigation purposes. Figure 3 is a topology for the testing environment.

D. System Testing and Results

The steps taken in this experimental research are performed as follows:

1. Mitigation testing with dummy data using REST API

This test is used to prove the functionality and measure the average response time of the core mitigation system. The Dynamic Access Control List algorithm is used in this experiment.

This test involves sending the data to the router by sending the data in the form of JSON REST API to configure the addition of the ACL rule, which is used to mitigate detected incoming positive attacks. The test was carried out 50 times by sending dummy data to routers. The obtained results will be calculated as average.

2. Testing of the DoS Attack detection and mitigation system

The testing phase of the detection system is performed by attacking the servers, as shown in the topology in Figure 3, according to the attack scenario for testing purposes. In this phase, the DoS attack was performed by Hping3, which generates malicious packets to flood the target system. The traffic generated from the attack resulted in self-generated data that the detection system will analyze. If the system detects a positive attack, the detection system will send the alert data to the mitigation system, which response by mitigating the attacker by automating the router.

The detection system will repeatedly analyze the collected traffic data every 45 to 55 seconds. Each scenario dataset consists of 15 minutes of collected traffic data.

There are six attack scenarios, with each connection or attack scenario consists of 15 minutes of attack or connection. All these scenarios were performed at different times. Table 3 shows the performed attack scenario in this experiment, along with the targeted server and port or service.

TABLE III. ATTACK SCENARIO TABLE

No.	Scenario	Description	Sampling Interval
1	Normal	2 clients to 1 server (Attacker 1 to Server 1, ICMP Ping)	45 seconds
2	DoS	1 client to 1 server, TCP SYN (Attacker 1 to Server 1, Port 80)	45 seconds
3	DoS	2 clients to 1 server (Attacker 1 - ICMP Flood to Server 1, Attacker 2 - TCP SYN Port 80 to Server 1)	45 seconds
4	DoS	1 client to 1 server, (Attacker 1 to Server 1, ICMP Flood)	45 seconds
5	DoS	2 clients to 1 server, (Attacker 1 and 2 to Server 1 - TCP SYN Flood Port 80)	45 seconds
6	DoS	1 client to 1 server, (Attacker 1 with Random IP to Server 1 - TCP SYN Port 80)	55 seconds

III. RESULT AND DISCUSSION

This chapter discusses in detail system evaluation such as the response time of the mitigation system, action taken by the system, and the discussion of the mitigation system.

The response time testing was used to measure the average response time of the core mitigation system. It is calculated from the time of each of the executed tests. Response time testing was performed by sending POST requests to modify Access Control Lists configuration to the routers 50 times. The testing scenario was performed by sending the request to a single router and sending the request to two routers. Table 4 shows the minimum, median, average, and maximum time elapsed for mitigation.

TABLE IV. MITIGATION RESPONSE TIME

Criteria	Time elapsed (1 router)	Time elapsed (2 routers)
Minimum	0,87 seconds	2,19 seconds
Median	1,45 seconds	3,26 seconds
Average	1,57 seconds	3,52 seconds
Maximum	3,84 seconds	8,58 seconds

Based on the obtained result, the mitigation system took an average time of 1,57 seconds of applying Access control list configuration in a single router, consists of 3 API requests for each incoming log, including security access token request, configuration retrieval, and applying the new configuration. Figure 5 shows the graph of average elapsed time to mitigate the attacks by implementing the Dynamic Access Control List method.

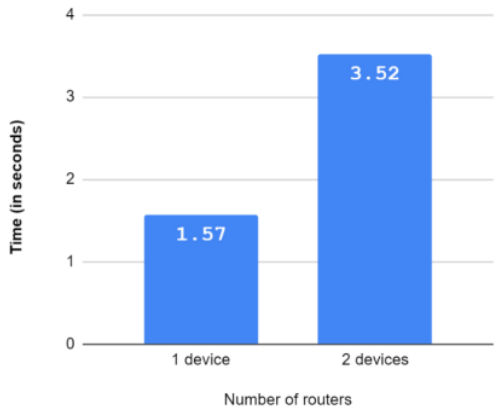

Figure 5 Average mitigation system elapsed time test chart diagram

The test result shows that the core of the mitigation system could execute the desired action by applying a dummy Access Control List configuration into the router.

The implementation of the detection system is tested based on the scenario shown in Table 3, with the result of the detection system testing implemented in the experiment for each scenario performed is as shown in Table 5.

TABLE V. DETECTION SYSTEM RESULT IN EACH SCENARIO

No.	Scenario	TP	FP	TN	FN
1	2 clients to 1 server (Attacker 1 to Server 1, ICMP Ping)	0	0	26	0
2	1 client to 1 server, TCP SYN (Attacker 1 to Server 1, Port 80)	0	0	0	20
3	2 clients to 1 server (Attacker 1 - ICMP Flood to Server 1, Attacker 2 - TCP SYN Port 80 to Server 1)	0	0	0	21
4	1 client to 1 server, (Attacker 1 to Server 1, ICMP Flood)	0	0	0	26
5	2 clients to 1 server, (Attacker 1 and 2 to Server 1 - TCP SYN Flood Port 80)	0	0	0	12
6	1 client to 1 server, (Attacker 1 with Random IP to Server 1 - TCP SYN Port 80)	0	0	0	8

The measurements of the algorithm implementation performance were using the Confusion Matrix method [11], which will be used to calculate the accuracy, precision, sensitivity. Table 6 shows the result of the implemented detection system for detecting Denial-of-service attacks in the Confusion Matrix table.

TABLE VI. DETECTION SYSTEM RESULT IN CONFUSION MATRIX

	Predicted Positive	Predicted Negative
Actual Positive	True Positive 0	True Negative 26
Actual Negative	False Positive 0	False Negative 87

The values of the accuracy, precision, sensitivity, are calculated from the total count of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP refers to correct predictions of attacks, TN indicates the normal data classified correctly as regular traffic. FP refers to incorrect predictions of attacks, and FN indicates the attack classified incorrectly as normal data. The result is as shown in the Table 6.

Sensitivity/True Positive Rate (TPR) measures the percentage of correctly identified attacks over the actual attacks in sampled traffic that are calculated using Equation (1).

$$Sensitivity = TP / (TP + FN) \quad (1)$$

Accuracy measures the percentage of proper detection over the sampled traffic calculated using Equation (2).

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (2)$$

Specificity/True Negative Rate (TNR) measures the system's ability to correctly detect sampled traffic without attack, is calculated by Equation (3).

$$Specificity = TN / (TN + FP) \quad (3)$$

The results of the detection system tests are in the form of a score table for Sensitivity, Accuracy, and Specificity, as shown in Table 6.

TABLE VII. DETECTION SYSTEM ANALYSIS RESULT

Criteria	Value in %
Sensitivity (TPR)	0%
Accuracy	23.01%
Specificity (TNR)	100%

The implemented detection system detects nothing but negative results, either when the attack occurs or when regular traffic occurs. We obtained 0% of Sensitivity (True Positive Rate), 23.01% of Accuracy, and 100% of Specificity (True Negative Rate), as shown in Table 7. It indicates that the detector cannot detect Denial-of-Service attacks since there are no positive results, resulting in no response from the mitigation system.

IV. CONCLUSION

Based on the research and testing performed, we can conclude that the mitigation system by automating REST API configurations could send dummy ACL requests to the router, with an average of 1,57 seconds response time for one router and 3,52 seconds for two routers. The detection system based on the Dynamic Thresholding algorithm is considered not usable, causing the whole mitigation system unusable.

The system evaluation shows that the DoS detection system used has a sensitivity (TPR) value of 0%, accuracy value of 23.01%, and specificity (TNR) value of 100%, without any positive attack results detected. As a result, the detection system did not trigger any alert request to the mitigation system that caused no mitigations taken. Further research suggests implementing other DoS attack detection algorithms and increasing the number of DoS attack scenarios.

ACKNOWLEDGMENT

This research was financially supported by The Faculty of Engineering, Universitas Diponegoro, through Strategic Research Grant 2021 number: 3178/S/komputer /2/UN7.5.3.2/PP/2021.

REFERENCES

[1] Cisco Systems, "Cisco Annual Internet Report (2018–2023)," *Computer Fraud & Security*, 2020. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed Feb. 13, 2021).

[2] A. F. Rochim, A. Rafi, A. Fauzi, and K. T. Martono, "As-RaD System as a Design Model of the Network Automation Configuration System Based on the REST-API and Django Framework," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 5, no. 4, pp. 291–298, Nov. 2020, doi: 10.22219/kinetik.v5i4.1093.

[3] R. Adhyatmaka Wiryawan and N. Rohman Rosyid, "Website-Based Network Administration Automation Application Development Using Python Programming Language, vol. 10, no. 2, pp. 741–752, Nov. 2019, Accessed: Jan. 21, 2021. [Online]. Available: https://jurnal.umk.ac.id/index.php/simet/article/view/3589.

[4] I. Karanta and M. Rautila, "An expert system for mitigation actions," in *Conference of Open Innovation Association, FRUCT*, Oct. 2017, vol. 2017-April, pp. 125–130, doi: 10.23919/FRUCT.2017.8071302.

[5] A. F. Rochim, M. A. Aziz, and A. Fauzi, "Design Log Management System of Computer Network Devices Infrastructures Based on ELK Stack," in *ICECOS 2019 - 3rd International Conference on Electrical Engineering and Computer Science, Proceeding*, Oct. 2019, pp. 338–342, doi: 10.1109/ICECOS47637.2019.8984494.

[6] J. Ramprasath and V. Seethalakshmi, "Secure access of resources in software-defined networks using dynamic access control list," *Int. J. Commun. Syst.*, vol. 34, no. 1, p. e4607, Jan. 2021, doi: 10.1002/dac.4607.

[7] S. K. Yadav, K. Sharma, and A. Arora, "Security Integration in DDoS Attack Mitigation Using Access Control Lists," *Int. J. Inf. Syst. Model. Des.*, vol. 9, no. 1, pp. 56–76, 2018.

[8] J. David and C. Thomas, "Efficient DDoS flood attack detection using dynamic thresholding on flow-based network traffic," *Comput. Secur.*, vol. 82, pp. 284–295, May 2019, doi: 10.1016/j.cose.2019.01.002.

[9] S. M. T. Nezhad, M. Nazari, and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks," *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 700–703, Apr. 2016, doi: 10.1109/LCOMM.2016.2517622.

[10] R. Rajendran, S. V. N. Santhosh Kumar, Y. Palanichamy, and K. Arputharaj, "Detection of DoS attacks in cloud networks using intelligent rule based classification system," *Cluster Comput.*, vol. 22, no. 1, pp. 423–434, Jan. 2019, doi: 10.1007/s10586-018-2181-4.

[11] M. E. Elhamahmy, H. N. Elmahdy, and I. A. Saroit, "A New Approach for Evaluating Intrusion Detection System," *CiiT Int.*, vol. 2, no. 11, pp. 290–298, 2010.

Design and Implementation of Post-Detection of Denial of Service (DoS) as a Mitigation System (PDDMS) Based on Dynamic Access Control List Algorithm

ORIGINALITY REPORT

7%	%	7%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Ike Pertiwi Windasari, Luqman Setyo Nugroho, Adian Fatchur Rochim, Risma Septiana. "An-SPf: as an Alternative Architecture of no Single Point Failure of Scalable Transcoding System Based on Kubernetes", 2021 International Conference on Computer System, Information Technology, and Electrical Engineering (COSITE), 2021 Publication	2%
2	"Paper Titles", 2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), 2021 Publication	2%
3	Jisa David, Ciza Thomas. "Efficient DDoS Flood Attack Detection using Dynamic Thresholding on Flow-Based Network Traffic", Computers & Security, 2019 Publication	1%

4	Information Management & Computer Security, Volume 21, Issue 4 (2013-10-19) Publication	<1 %
5	Xuefeng Xian, Pengpeng Zhao, Wei Fang, Jie Xin, Zhiming Cui. "Automatic classification of deep web databases with simple query interface", 2009 International Conference on Industrial Mechatronics and Automation, 2009 Publication	<1 %
6	Akira Yamashita, Wataru Muro, Masayuki Hirono, Takehiro Sato, Satoru Okamoto, Naoaki Yamanaka, Malathi Veeraraghavan. "Hadoop triggered opt/electrical data-center orchestration architecture for reducing power consumption", 2017 19th International Conference on Transparent Optical Networks (ICTON), 2017 Publication	<1 %
7	Hongbin Luo, Zhe Chen, Jiawei Li, Thanos Vasilakos. "On the Benefits of Keeping Path Identifiers Secret in Future Internet: A DDoS Perspective", IEEE Transactions on Network and Service Management, 2018 Publication	<1 %
8	Xinqian Liu, Jiadong Ren, Haitao He, Qian Wang, Chen Song. "Low-rate DDoS attacks detection method using data compression	<1 %

and behavior divergence measurement", Computers & Security, 2021
Publication

9	Cagatay Catal. "Software Fault Prediction with Object-Oriented Metrics Based Artificial Immune Recognition System", Lecture Notes in Computer Science, 2007 Publication	<1 %
10	"MCSE 70-293: Self Test Questions Answers and Explanations", Elsevier BV, 2003 Publication	<1 %
11	Everton Leonardo Skeika, Mathias Rodrigues Da Luz, Bruno Fernandes, Hugo Siqueira, Mauren Louise Sguario Coelho De Andrade. "Convolutional Neural Network to detect and measure fetal skull circumference in ultrasound imaging", IEEE Access, 2020 Publication	<1 %
12	Veena H. Bhat, Krishna S., P. Deepa Shenoy, Venugopal K. R., L. M. Patnaik. "Chapter 13 JPEG Steganalysis Using HBCL Statistics and FR Index", Springer Science and Business Media LLC, 2010 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

Design and Implementation of Post-Detection of Denial of Service (DoS) as a Mitigation System (PDDMS) Based on Dynamic Access Control List Algorithm

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6