International Journal of Information Sciences and Engineering | ***ISSN 1694-4496***
Published by AIR JOURNALS | *https://airjournal.org/ijise*
16/18 Avenue des Longaniers, Quatre Bornes, Mauritius
airjournals@gmail.com; enquiry@airjournal.org

SCHOLARLY ARTICLE

Development of Java Logical Program Synchronous Floods Distributed Denial of Service Algorithm, for Improving Data Security at Backend Server

[1]Okafor P. U., [2]Arinze S. N., [3]Uka C.
[1,2,3]Enugu State University Of Science And Technology, Enugu State, Nigeria

Accepted: September 16th, 2022	Published: September 30th, 2022

Citations - APA

Okafor P. U., Arinze S. N. & Uka C. (2022). Development of Java Logical Program Synchronous Floods Distributed Denial of Service Algorithm, for Improving Data Security at Backend Server. *International Journal of Information Sciences and Engineering, 6(2), 1-10. DOI: https://doi.org/10.5281/zenodo.7151791*

ABSTRACT

The work aims to improve data security at the backend server using the SYN Floods detection algorithm (Java Logical Program). In the development of the Java Logical Program (JLP), Java programming and Hypertext scripting languages were used in the development of the anomaly detection algorithm, while Hypertext Markup Language was used for the deployment of the system using the NetBeans integrated development environment. A TCP 3-Way Handshake connection protocol was deployed in establishing a full-duplex communication between the client and the server in the network. MySQL and SQLite were deployed to create the database connection and objects for an easy query of the database. The functionality test was carried out based on the data rate threshold and time of response. Test deployed flow records from WIDE MAWI WORKING GROUP repository for the developed JLP, SYN and FIN Difference (SynFinDiff) and Partial Completion Filter (PCF), two methods for monitoring and analyzing network traffic in detecting SYN flooding attacks. Results showed that JLP achieved a traffic data rate threshold of 11 as against SynFinDiff and PCF thresholds of 5 and 20 respectively. The significance of the result is that the developed system will not be reporting an attack for any period during which traffic is under the threshold thereby, eliminating false positives attack when TCP retransmission packets with half-opened connections are detected. Also, JLP detected an attack on data stored at the backend server within 0.17 seconds, which is a 43% improvement compared to 0.3seconds achieved by SynFinDiff which on the other hand outperformed the PCF response time.

Keywords: *Denial of Service Algorithm; Backend Server; Data Security; Partial Completion Filter (PCF); SynFinDiff (SYN and FIN Difference)*

Introduction

In modern client-server applications in various industries, most of the sensitive data stored are consequently leaked on the backend. Web services and mobile applications provide convenient front-end mechanisms to access and manipulate the data stored in backend systems. Among the sensitive data are customers' personal information, identity, or access credentials, which typically constitute the greatest value for potential attackers. The adverse effect is quite detrimental when this information is in the wrong hands. Unsurprisingly, front-end applications become a target for adversaries seeking a way to gain access to backend systems and the databases they contain. These adversaries can gain access to the data in the backend server through: altering the front-end behavior, sniffing traffic between the database and the front-end application, altering the behavior of the database to bypass the access, stealing the assets from the files by accessing the database host directly and getting the database files at rest and extracting the meaningful data from them, etc.

The backend is responsible for an application's business logic and calculations and for storing and retrieving data. The backend usually consists of a backend application (or backend script), a server, a database, and Application Programming Interfaces (APIs) (Haitao et al., 2013). With the backend, industries can use services and data stored at any physical location outside their control (TeskaLabs, 2015). This facility raised various security questions like privacy, confidentiality, integrity, authentication, etc., and demanded a trusted computing environment wherein data confidentiality can be maintained.

Literature Survey

Denial of Service (DOS) attacks is popular due to their ability to significantly affect networks. DOS, as the name implies, exhausts the resources of the target network by sending invalid traffic. A DOS attack when carried out by multiple devices is known as a Distributed Denial of Service (DDOS) attack (US-CERT, 2013). DDOS is considered to be the biggest threat to networks.

SynFinDiff (SYN and FIN Difference) Method: The method of detecting SYN flooding attacks called Synfindiff was first introduced by (Wang et al., 2002). The authors describe its advantages in low computational overhead and above all in statelessness, which makes it a method that is itself resistant to SYN flooding. The detection mechanism is based on monitoring SYN-FIN (or RST) TCP flags during communication. The algorithm belongs to the group called Sequential Change Point Detection Algorithms (Basseville & Nikiforov, 1993). In statistical analysis, these methods are used to identify changes in the studied stochastic processes or time series. To make the mechanism easier to implement and more generally applicable, the authors use the CUSUM (Cumulative Sum) method.

Partial Completion Filter (PCF) Method. The Partial Completion Filter (PCF) was introduced in (Kompella et al., 2007). Here, partial completion indicates the state of an open but unfinished TCP session. In addition to SYN flooding attacks, it also detects claim-and-hold attacks that create a lot of server sessions, and complete the synchronization process, but there is no activity in the created session. Each process created in this way must then wait until the timeout expires and takes up the computation sources for legitimate connections. The PCF, therefore, focuses on monitoring a large number of admissions SYN without terminating FIN packets.

Java Logical Program (JLP) Algorithm. This anomaly detection algorithm detects attacks that have an anomalous behavior from the normal behavior of Transport Control Protocols (TCP) characteristics which state that the client sends the first SYN segment to the server, after receiving the SYN segment from the client; the server sends an SYN+ACK segment back to the client and then client responds with ACK segment and the connection is established. It reduces to the barest minimum, the false positive alarm when TCP retransmission packets with half opened connection are detected

Methodology

Hypertext Markup Language was used for the implementation of the JLP using the NetBeans integrated development environment. A TCP 3-Way Handshake connection protocol was designed and implemented using sequence and acknowledgment numbers to establish full-duplex communication between the client and the server in the communication network. The communication network is set up using Ethernet LAN with CISCO equipment. MySQL

and SQLite were deployed in the creation of the database connection and objects through programming for easy querying of the database

Two methods for monitoring and analyzing network traffic, used in detecting SYN flooding attacks namely; SynFinDiff (SYN and FIN Difference) and PCF (Partial Completion Filter), were characterized to determine their data rate threshold and time response. The result is used to compare the improvement of the proposed method. The methods were chosen because each of them possessed certain similar characteristics. The primary similarity is that each is intended to detect the same kind of attack; SYN flooding. Each is designed to be deployed at the edge of a leaf network and uses a constant amount of state, operating on a time scale of tens of seconds. TCP control packets (SYNs, SYN/ACKs, FINs, and RSTs) were used as its input.

JLP Algorithm

The SYN Floods Anomaly detection architecture is such that it takes in input files (traffic packet trace), perform SYN floods anomaly detection, stores the detected attacks, and outputs the result. Furthermore, the system would export the detected attacks using Excel CSV export algorithm (java program code). The user would have the privilege to quarry the database. The algorithm for SYN Floods DDoS Attacks detection is shown in Table 1.

Table 1: Anomaly Detection Algorithm

S/N	Algorithm
1.	**Database connection initiated**
2.	**Input** {packet traces}
3.	Catch and Handle Exceptions
4.	Authenticate input variables
5.	**Store** 'packet traces'
6.	**Index, Sort, Search** 'Attack Record' **and** 'Detect anomalies'
7.	**Update** 'Attack record'
8.	**Output** {detected anomalies}
9.	**Database connection closed;**

This work is deployed using Transport Control Protocol (TCP) network, which is the major network protocol that attackers mostly use to launch SYN floods attacks on the targeted server. In judging the proposed anomaly detection algorithm, the judgment will be based on the TCP three-way handshake in Transmission Control Protocols, which is the method used by TCP to set up a TCP/IP connection over an internet protocol-based network. The detected attacks must have the following criteria;

1. A SYN floods attack inundates a site with [SYN] segment containing forged (spoofed) IP source addresses with non-existent or unreachable addresses.
2. The host server responds with [SYN, ACK] segments to these addresses and then waits for responding acknowledgment [ACK] segments.
3. The host server will not receive any acknowledgment [ACK] response and eventually time out. This is because, in criterion No.2, the response was sent to non-existent or unreachable IP addresses.
4. By flooding a host with an incomplete TCP connection ([SYN] and [SYN, ACK] without an [ACK]), the detection algorithm result must show that the attacker eventually attempts to fill the memory buffer of the victim. Note that once this buffer is full, the host can no longer process new TCP connection requests. The flood might even damage the victim's operating system. Either way, the attack disables the victim and its normal operations.
5. The attacker must send a request with multiple IP addresses of the same Port Number against a Victim's IP address with varying Port numbers or a single Port Number.
6. The attack duration helps to know how harmful the attacks were to the victim. This means that attacks became harmful when it occurs within a few minutes and more harmful when it occurs within a few seconds.

The JLP detection algorithm is expected to detect attacks with the above criteria. The flow chart for the anomaly detection algorithm is shown in Figure 1.

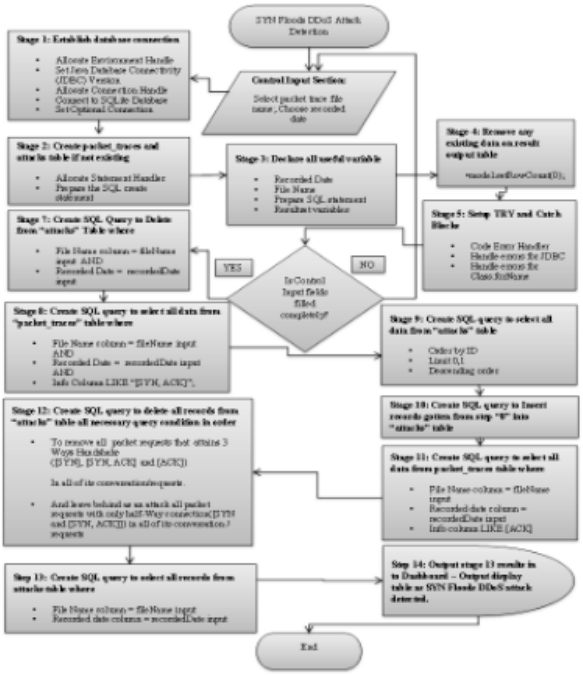

Fig 1: Anomaly Detection Algorithm

Modelling the System Database

This is sketching out the schema, which is developing the entity relation diagram (ERD) (Uka et al., 2022). This application exploits the Entity-Relationship (E-R) model in establishing the relationships of the objects. This is vital because it makes query creation easy and the joining conditions valid. This application also uses the Visual Paradigm community edition to achieve the sketching. Figure 2 shows the industrial system and anomaly detection system, database model.

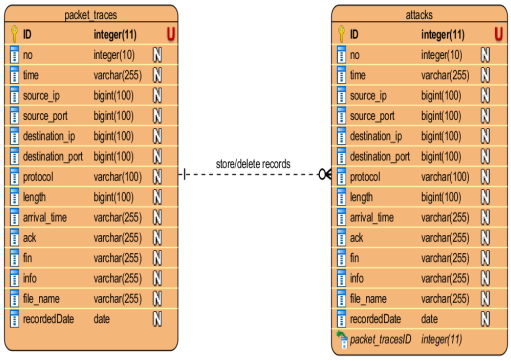

Fig 2: Anomaly Detection Algorithm ERD Model.

Detection of SYN Floods DDoS attack

This is where the JLP anomaly detection algorithm for network traffic packets is applied. The performance of the SYN Floods detection submenu depends on the control input parameter. The control input parameters are defined by selecting the packet trace name and recorded date for the trace and submitted for the execution of the SYN Floods

detection task. The algorithm looks into the database Packet_traces table record and fetches records where the selected control input parameters and INFO column are SYN and ACK. If matches were found, then the next lines of code store all match results into Attacks_table.

Subsequently, a record deleting task from Attacks_table is performed. It deletes any row/record where ACK is found inside the INFO column using Source IP, Source Port, Destination IP, and Destination Port. Or delete any row/record where ACK is found inside INFO column using Destination IP and Destination Port, Destination IP and Destination Port columns matching with their respective column associated values fetched from Packet_traces table where ACK exists in INFO column. Then, these records are used to delete from the Attacks_table record where this value exists. These deleted records are those network packets that have TCP 3-Ways Hand Shake connection. In other words, the deleted records are legitimate TCP packet connections. At this stage, the remaining packet records in the Attacks database table are now seen as SYN Floods Distributed Denial of Service attack detected. The detected attack results are displayed on the Output_table.

Detected Attack History – This is another submenu under the DDoS attack submenu. Its function is to display on the dashboard – Output section table, the already detected SYN Floods attacks existing in the Attacks database table, concerning the specified control input parameters for a particular network packet trace record.

Chart Representation of SYN Floods Attacks Detected – This submenu, when clicked, makes use of the selected control input parameter, such as the selection of packet trace name and the captured date of the trace selected to perform data analysis and represent the number of attacks detected in a bar chart.

SYN Floods Attacks Summary – This submenu provides the summary of all SYN Floods attacks detected by grouping the attacks based on each packet trace and captured date of each network packet trace.

Test and Result
Testing the Packet Trace File Upload

To test the packet trace file uploading functionality, Flow records were used, which were collected from the MAWI working group repository (MAWI Working Group Traffic Archive, 2017) directory. The publicly available packet traces were recorded from November 15, 2017, through November 24, 2017. The network datasets used are 10 in number and were captured within the same period. The original files are in *pcap* extension format. The algorithm was able to analyze them separately based on their captured data. The total number of network packet samples used is 494,122 TCP data.

The upload section was tested and the results are shown in Figures 3 and 4. The uploaded data are displayed on the dashboard – Output_table and task completed alert prompts. The packet uploaded has the following: No, Time, Source IP, Source Port, Destination IP, Destination Port, Protocols, Arrival Time, Length, Acknowledgement, Finish, and Info column as the data inputs to the upload section. All these columns are stored in the database Packet_trace table, from which every other analysis is performed.

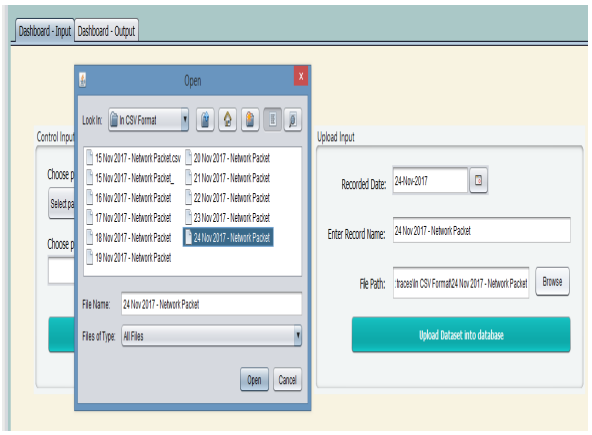

Fig 3: Packet Traces File Uploading Process

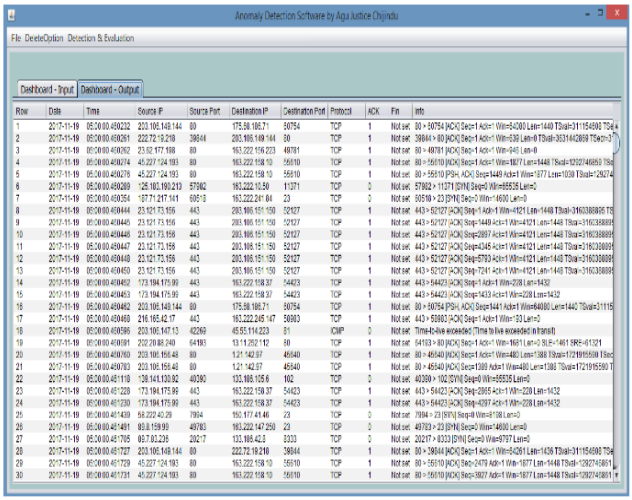

Fig 4: Successful Upload of a Packet Trace File into Database.

Evaluation of SYN Floods Attacks Detected in Traffic Traces

The SYN Floods detection algorithm was tested on each of the network packets traces data in the database Attacks_table and then analyses the data for representation in a bar chart. Four-day network traffic packet trace was found SYN Floods attack out of ten-day capture. The detected SYN Floods attacks in each of the affected network packet traces captured from 15th – 24th November 2017 are displayed on the Dashboard. The following Destination IP addresses "58.34.195.74" and "202.235.207.10" are the victims that received "44" and "19" SYN floods attacks respectively. The distributed denials of service attack (DDoS) attackers sent multiple numbers of spoofed packets using different source IPs with the same source port number "22", targeting one particular host server (destination IP). These were mostly experienced in IP address "58.34.195.74". These attacks consequently vary the range of the host server ports which exhaust the resources of the target server and deny the legitimate user the opportunity to connect. The attacks on "58.34.195.74" started at 05:00:00.270247 and ended at 05:00:01.111302.

The DDoS attackers sent "16" spoofed packets using different source IP addresses with source ports number "443" and "80", targeting one particular host server "202.6.253.132". Thereby varies host server ports with the aim to exhaust the resources of the server and deny the legitimate user the opportunity to connect.

Result and Discussion

The summary of all the attacks detected was presented in a simple graph for packet traces captured from 15th Nov 2017 to 24th Nov 2017. Figure 5 and Table 2 shows how the Anomaly Detection algorithm performed in detecting SYN Floods DDoS attack in any given network traffic packet.

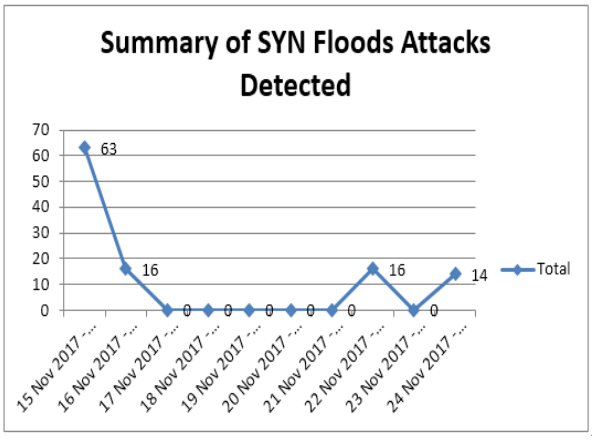

Fig 5: Summary of SYN Floods Attacks Detected from Packet Traces Captured on 15th Nov 2017 through 24th Nov 2017

Graph of Figure 5 shows the packet trace datasets on the horizontal axis and the total number of attacks in each packet trace dataset on the vertical axis. It displays the summary of SYS Floods DDoS attacks detected in each of the network packet traces. It can be observed that attacks were detected in only four datasets, which are network packet traces captured on the 15th.

Table 2, presents the summary of the entire network traffic datasets used. It shows the packet trace names and the number of network packets or conversations between client and host servers in each dataset. It also presents the number of attacks and victims detected using the proposed JLP.

Table 2: Detection Summary of Packet Traces Captured from 15th Through 24th November, 2017.

S/N	PACKET TRACE NAME	TCP Packets	No. of Attacks	Attack Duration	Frequency of an Attack Sent in Seconds	No. of Victims
1.	15 Nov 2017 – Network Packet	84,636	63	0.84, 0.88	0.2, 0.05	2
2.	16 Nov 2017 – Network Packet	45,977	16	0.35	0.02	1
3.	17 Nov 2017 – Network Packet	32,806	-	-	-	-
4.	18 Nov 2017 – Network Packet	22,440	-	-	-	-
5.	19 Nov 2017 – Network Packet	48,163	-	-	-	-
6.	20 Nov 2017 – Network Packet	52,900	-	-	-	-
7.	21 Nov 2017 – Network Packet	65,107	-	-	-	-
8.	22 Nov 2017 – Network Packet	48,423	16	0.49	0.03	1
9.	23 Nov 2017 – Network Packet	48,721	-	-	-	-
10.	24 Nov 2017 – Network Packet	44,949	14	0.17	0.01	1

The result obtained using the proposed JLP SYN Floods DDoS attack detection algorithm, shows that SYN floods attack inundates a host server with [SYN] segment containing forged (spoofed) IP source addresses with non-existent or unreachable addresses. The host server responds with [SYN, ACK] segments to these addresses and then waits for responding acknowledgment [ACK] segments. The host server will not receive any acknowledgment [ACK] response and eventually time out. This is because the response was sent to non-existent or unreachable IP addresses. JLP algorithm detects those attacks flooding a host with an incomplete TCP connection ([SYN] and [SYN, ACK] without an [ACK]). The detection algorithm result shows that the attacker eventually attempts to fill the memory buffer of the victim. Once this buffer is full, the host can no longer process new TCP connection requests. This flood might even damage the victim's operating system. Either way, the attack disables the victim and its normal

operations. Results show that the attacker sent a request with multiple IP addresses of the same Port Number against a Victim's IP address with varying Port numbers or a single Port Number.

In Figure 6, the threshold used in JLP, SynFinDiff algorithms has a lot to contribute to the body of knowledge, many of the false alarms and minor anomalies were seen by SynFinDiff, spring from small increases in traffic during periods of light network activity.
This is because it uses a threshold traffic rate of "5" below which any supposed attack is irrelevant. The developed Anomaly Detection (JLP) algorithm achieving a traffic rate threshold of 11 is a fairly conservative figure because it is not reporting an attack for any period during which traffic is under the threshold thereby eliminating false positives attack.

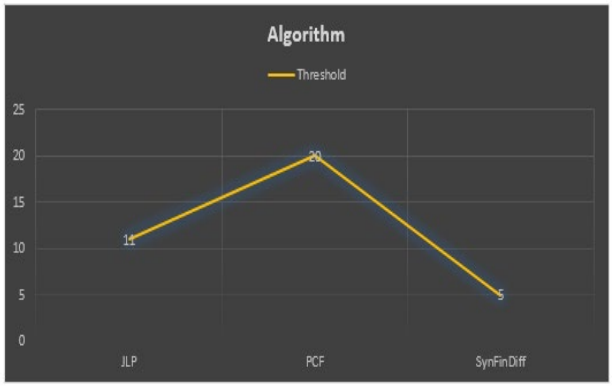

Fig 6: Threshold Evaluations of SynFinDiff and JLP Algorithms

Java logical Program (JLP) detected two attacks in trace 1 and one attack in trace 2, trace 8 and trace 10 respectively. This is considered as an absolute attack with zero false positive attack detection alerts, due to the threshold of 11 SYNs that was set, below which any supposed attacks are irrelevant. Java Logical Program (JLP) algorithm reduces to the barest minimum the false positive alarm when TCP retransmission packets with half opened connection are detected. Such requests are ignored for a period during which their traffic is below the threshold and should not be considered an attack.

From Figure 7, the JLP algorithm detected attacks that have an anomalous behavior from the normal behavior of TCP characteristics which states that firstly, the client sends the first segment to the server, and after receiving the SYN segment from the client; the server sends an SYN+ACK segment back to the client and then client responds with ACK segment and the connection is established.

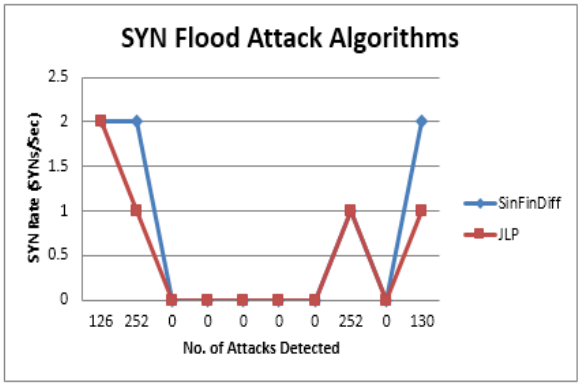

Fig 7: SYN Floods Attack Detection Performance Evaluation.

In Figure 8, $SynFinDiff$ algorithm shows an essentially flat response time over most of the range tested, with $SynFinDiff$ response time of about 3 seconds slower than JLP. The JLP takes 0.17 seconds to respond to the smallest (14 SYN/s) attack, versus 0.49 to 0.88 seconds for any higher rate. The response time of JLP is more efficient compared to $SynFinDiff$

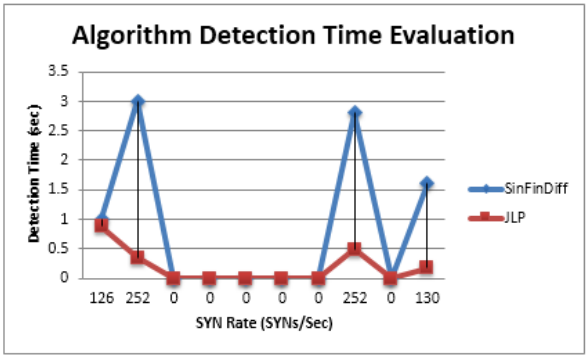

Fig 8: Algorithm Detection Time Evaluation

Conclusion

The Anomaly Detection Algorithm application was executed without errors. It could only generate an error where the wrong datatype input, authentication, or authorization is made. There were no error messages from the Netbeans integrated development environment, and that implies that there were no error messages from the Netbeans integrated development environment and that means that the application runs but do not prove its functionality and interoperability.

The JLP algorithm achieved a traffic data rate threshold of 11 as against the SynFinDiff and the PCF thresholds of 5 and 20 respectively. Achieving a traffic rate threshold of 11 implies that the algorithm will not be reporting an attack for any period during which traffic is under the threshold thereby eliminating false positives attack when TCP retransmission packets with half opened connection are detected. The system detected an attack within 0.17 seconds which is a 43% improvement in time response when compared with the SynFinDiff algorithm. Based on the results achieved, it can be concluded that with the improved backend server security model, data can be transferred from client to server ends via an unsecured network and an eavesdropper that breaks into the message will return a meaningless message also, the system will not be raising a false alarm on an attack for any period during which traffic is under the threshold.

Acknowledgement
The authors wish to thank the Tertiary Education Trust Fund (TETFUND), Nigeria for sponsoring this project

References

Basseville, M., and Nikiforov, I. V. (1993). Detection of Abrupt Changes: Theory and Application. PTR Prentice-Hall, Inc., Englewood Cliffs.

Haitao W., Zhenqian F., and Chuanxiong G., (2013). ICTCP: Incast Congestion Control for TCP in Data-Center Networks. *IEEE/ACM Transactions on Networking*, 21(2).

Kompella, R. R.; Singh, S.; Varghese, G., (2007). On Scalable Attack Detection in the Network. *IEEE/ACM Trans*. ISSN 1063-6692.

MAWI Working Group Traffic Archive. http://mawi.wide.ad.jp/mawi/. Accessed Nov 2017.

TeskaLabs (2015). Understanding the Importance and Value of Backend Security. Retrieved from https://teskalabs.com/blog/backend-security-importance. Last accessed: 10-06-2021.

Uka C., Eneh I. I., and Okafor P. U., (2022). Development of security solution for back-end server in a network environment using a digital signature with rivest-shamir-adleman (rsa) algorithm. *Irish International Journal of Engineering and Applied Sciences,* 6(01), 13-23.

US-CERT, (2013). Understanding Denial-of-Service Attacks. Retrieved from: https://us-cert.cisa.gov/ncas/tips/ST04-015. Last Accessed: 10-06-2021

Wang, H., Zhang, D., Shin, K., (2002). Detecting SYN Fooding attacks. In INFOCOM 2002.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, roèník 3, ISSN 0743-166X, Pp 1530 -1539.