# Slow Hypertext Transfer Protocol Mitigation Model in Software Defined Networks

Opeyemi Aderiike Abisoye
*Department of Computer Science*
*Federal University of Technology, Minna*
Niger State, Nigeria
o.abisoye@futminna.edu.ng

Oluwatobi Shadrach Akanji
*Department of Computer Science*
*Federal University of Technology, Minna*
Niger State, Nigeria
akanjioluwatobishadrach@yahoo.com

Blessing Olatunde Abisoye
*Department of Computer Engineering*
*Federal University of Technology Minna*
Niger State, Nigeria
b.abisoye@futminna.edu.ng

Joseph Awotunde
*Department of Computer Science*
*University of Ilorin*
Kwara State, Nigeria
awotunde.jb@uniilorin.edu.ng

***Abstract*—Distributed Denial of Service (DDoS) attacks have been one of the persistent forms of attacks on information technology infrastructure connected to a public network due to the ease of access to DDoS attack tools. Researchers have been able to develop several techniques to curb volumetric DDoS attacks which overwhelms the target with large number of request packets. However, compared to volumetric DDoS, low amount of research has been executed on mitigating slow DDoS. Data mining approaches and various Artificial Intelligence techniques have been proved by researchers to be effective for reduce DDoS attacks. This paper provides the scholarly community with slow DDoS attack detection techniques using Genetic Algorithm and Support Vector Machine aimed at mitigating slow DDoS attack in a Software-Defined Networking (SDN) environment simulated in GNS3. Genetic algorithm was employed to select the features which indicates the presence of an attack and also determine the appropriate regularization parameter, C, and gamma parameter for the Support Vector Machine classifier. Results obtained shows that the classifier had detection accuracy, Area Under Receiver Operating Curve (AUC), true positive rate, false positive rate and false negative rate of 99.89%, 99.89%, 99.95%, 0.18%, and 0.05% respectively. Also, the algorithm for subsequent implementation of the selective adaptive bubble burst mitigation mechanism was presented.**

***Keywords—genetic algorithm, slow DDoS mitigation, slow distributed denial of service, software defined network, support vector machine.***

## I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks are assaults against network, digital and information technology infrastructure which involves the use of Internet-enabled and connected devices to synchronously send either legitimate or illegitimate requests to the victim at a rate which overwhelms the processing and response rate of the victim [1], [2]. A large portion of DDoS attacks reported have been volumetric DDoS attacks that send large numbers of requests at a rate faster than the victim can process. Due to the ease with which volumetric DDoS attacks easily triggers control measures, attackers have resorted to using a form of DDoS that occupies resources in a manner that resembles the way a legitimate client would request for resources [3]–[5]. This form of DDoS is known as slow DDoS.

Slow DDoS or low-rate attacks [6], [7] are hard to detect and usually exploit the operation of the application layer. It exploits application layer protocols such as HTTP, Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), and Internet Message Access Protocol (IMAP) by behaving either as a legitimate client sending traffic over a slow connection or one with low response processing capacity [2, 7, 8]. Slow DDoS causes service unavailability by occupying all or most connections to the victim and sustaining the connection for a long time by sending data to the victim over the connections [9]. Attackers use variations of the slow DDoS, known as slow HTTP DDoS, to target web servers due to large number of web services on the Internet.

Slow HTTP DDoS attack is a type of slow DDoS targeted at web servers which is launched after a Transmission Control Protocol (TCP) connection has been established with the victim web server [9], [10]. There are three variations of slow HTTP DDoS: slow HTTP header, slow HTTP POST, and slow read DDoS attacks. Slow HTTP header DDoS attacks, also known as slow GET attacks, sends HTTP GET messages to the web server without transmitting two carriage return (CR) line feed (LF) which signifies the end of the GET request. In essence, the request from the client is not concluded which makes the web server wait indefinitely for the completion of the request before processing of the header can begin [5, 7, 10]. Slow HTTP POST DDoS attacks uses the *Content-Length* field in the header to inform the web server of a large *data* transfer. However, instead of sending the data at once, the attack focuses on sending the data in small chunks thus prolonging the connection to the web server [10], [11]. Unlike slow HTTP header and POST DDoS attacks which are based on data transmission from the attacker to the web server, slow read attacks are based on data transmission from the web server to the attacker. The slow read DDoS attacker requests for a resource on the server that has large data to be transmitted but adjusts the TCP window of the attack machine so as to force the web server to send the data in small bytes thus prolonging the connection time.

Mitigation of the slow HTTP DDoS attack refers to the use of methods that prevent service degradation or resource exhaustion on the web server [12] when an attack is detected by halting or diminishing the rate of attack [4]. Previous volumetric and slow DDoS attack mitigation techniques include redirecting the traffic to a verifying device [13]–[15], limiting the rate of attack [16], [17], dropping the attack traffic selectively [18], or spreading the traffic to replicas of the attack victim [19], [20]. A global view of the network status and traffic that traverses the

978-1-7281-9675-6/20/$31.00 ©2020 Crown

network is needed for effective detection and mitigation of slow DDoS [14].

The problem of a unified network view, management, and flexible device configuration for Software Defined Networking (SDN) is solved by fusing a logical centralized network management with network programmability through the separation of the data plane from the control plane [21]. SDN is comprised of network controller and switches. The controller, operating at the control plane, governs the manner of data forwarding and the switch, operating at the data plane, receives and forwards data based on rules defined by the controller [1]. Collection of traffic data is performed by the controller through the Openflow protocol. However, traffic data generation is not lightweight in Openflow when compared to netflow because the controller needs to request for the data at intervals from the switch. In netflow, the traffic data is exported without the request-response overhead [15], [22]. Netflow is a technology of Cisco Systems that monitors network traffic and exports the network flows. Network flow refers to the unidirectional network packet stream between source and destination applications [15] which offers efficient storage of network packets by grouping them into flow summaries [23]. Using netflow for traffic collection in SDN also reduces packet processing overhead compared to Full Packet Captures (FPC) [23].

In this work, we generate a dataset of attack and legitimate traffic from netflow version 5 record exports in a simulated SDN network in GNS3. Genetic algorithm is then used to select the appropriate features that determines the status of the traffic being examined with the selection of regularization and gamma parameters for the SVM classifier. The SVM classifier is executed on the final dataset using the selected regularization and gamma parameters to generate a model for real-time detection of attacks. Also, the selective adaptive bubble burst mitigation mechanism is proposed.

## II. LITERATURE REVIEW

Machine learning algorithms from supervised and unsupervised learning techniques were used in [24] to detect Denial of Service (DoS) attacks of slow POST and slow header (slowloris). 5-NN, JRIP, random forest, multilayer perceptron (MLP), naïve bayes, support vector machine, logistic regression, and C4.5 decision trees were learning algorithms employed in the work. High Area Under Reciever's Operating Curve (AUC) was recorded partly attributed to the use of Netflow for traffic collection. In the normal to attack traffic class ratio distribution of 50:50, random forest achieved the highest AUC of 0.99905 among other class distributions and learning algorithms. Detection of slow HTTP attacks using KNN, SVM, logistic regression, random forest, decision trees, and deep neural networks was examined in [25]. High detection accuracy of 99.87% and 99.81 % were achieved by decision trees and KNN respectively. This signifies that KNN, an unsupervised learning algorithm, can be used in detecting slow attacks effectively.

In [26], detection of slowloris and slow POST attacks in encrypted traffic. Single linkage clustering, k-means clustering, fuzzy c-means, self-organizing maps, and DBSCAN were the machine learning techniques used for detection. High detection rates of 99.9957% with false positive rate of 0.0043% for slowloris attacks was recorded in k-means, self-organizing maps, and fuzzy c-means. For slow POST attacks, k-means, self-organizing maps, and fuzzy c-means achieved high detection rates of 99.9931% with false positive rate of 0.0043%.

Analysis of the window size and packet inter-arrival times was used to detect slow HTTP DDoS attacks in [5]. Measurement of the stress on the web server was employed in [12] to detect slow HTTP attacks. Once the attack is detected, a reverse proxy mechanism handles all subsequent incoming traffic for the primary web server. Performance model based on central processing unit (CPU) utilization and time, workload , throughput, waiting time, and disk utilization of the web server to signify the presence of attack traffic was examined in [27]. However, the period to establish the baseline became a problem because it might be established when an attack is in progress or when all traffic use cases could not be captured.

Analysis of log files to establish similarity using Euclidean distance similarity metric was employed in [28]. Hellinger distance, a distance similarity metric, was used in [29] to measure the distance between the probability distributions of the attack and normal traffic generated. Detection evasion is expected if an attacker can create packets whose probability distribution is similar to that of the normal traffic used as a standard. Chi-square statistics was used to detect slow rate DoS attacks in [30]. Appropriate selection of threshold and interval time proved difficult.

Nonparametric CUSUM algorithm was applied in [31] to detect slow header, POST, and read DoS attacks. 13 sampling techniques that detects changes in the distribution of observed values were used. It was observed that as the threshold number increases, detection rate reduces. Detection rate of 100% was recorded when the threshold value reached 2500. When the rate of sampling exceeded 20%, selective flow sampling achieved the highest detection rate.

## III. PROPOSED GENETIC ALGORITHM WITH SUPPORT VECTOR MACHINE DETECTION TECHNIQUE

The use of machine learning algorithms to detect slow HTTP DDoS attacks have proven to be useful and more accurate than other methods. However, selecting the parameters of the classifier and determining the appropriate features to use in distinguishing a legitimate traffic from an attack traffic is of importance so as to improve the results to be obtained. In this work, support vector machines classifier was used to distinguish between legitimate and attack traffic. The support vector machine has two parameters: the regularization (C) and the gamma parameter which aid in constructing the optimal hyperplane between the data points. Genetic algorithm was used to select the combination of features that signify the presence of an attack. Also, genetic algorithm was used to tune the SVM parameters so as to obtain an optimal result.

The dataset used was generated in a GNS3 SDN simulation in which the switch exported netflow records obtained to the netflow collector resident on the controller after a timeout value

is reached. Through the netflow collector, the controller aggregated all the netflow records received. Slow header, POST, and read attacks were simulated using the slow http test tool which was selected because of the ease of configuration and tuning of attributes. Normal traffic was simulated in addition to legitimate slow clients whose traffic patterns bear similarity with slow header, POST, and read attacks. The legitimate slow clients' functionality was simulated using python.

## IV. PROPOSED SELECTIVE ADAPTIVE BUBBLE BURST MITIGATION TECHNIQUE

Selective adaptive bubble burst slow HTTP DDoS mitigation technique was derived from the synthesis of adaptive bubble burst DDoS mitigation technique in [20]. In the adaptive bubble burst technique, once attack is detected, all traffic is spread across all the replica web servers. However, in selective adaptive bubble burst mitigation technique, the traffic flagged as malicious is rerouted to the replica servers using a method that ensures the monitoring of the flagged traffic's behavior even when it was rerouted so as to block the traffic when necessary. The traffic is blocked when it violates the conditions defined relative to the number of replica web servers and the number of times the traffic is rerouted to another web server. Fig. 1 depicts the operation of this technique.

The Selective Adaptive Bubble Burst Mitigation Mechanism operations can be modelled as a function of the number and IP addresses of the web servers ($y_1...y_n$) within the network and the connection requests $\sigma_{ci(x)}$ (where $\sigma_c$ is the connection symbol, $i$ is the connection number, and $x$ is the client's IP address), attack or legitimate, sent to the primary web server ($y_1$). Here, $x$ is the client's source IP address and the netflow record of source IP $x$ is depicted by $N_{fx}$. The SVM model generated after classification of the dataset is denoted by $\theta$. A change in the value of destination IP address $y$ expressed as $y'$ occurs when the classification of $N_{fx}$, netflow record of $x$, gives a class category $N_c$ which is an attack. Once the number of times $N_c$ yields as an attack traffic causes $y'$ to equal the last replica server $y_n$, the block parameter $\mu_x$ for the defaulting IP address $x$ is propagated to the switch which blocks all traffic from IP address $x$.

### A. Selective Adaptive Bubble Burst Algorithm

**Input**: SVM Model $\theta$, Netflow record of IP source IP address $x$ is $N_{fx}$, primary webserver $y_1$, replica set of webservers $\{y_\pi\}$ where $\{y_\pi\} = \{y_2, y_3, \cdots, y_n\} : n \geq 2 \wedge n = |y_1 \cup \{y_\pi\}|$
**Output**: Boolean block value $\mu_x$ for IP address $x$
**Procedure** *onlineSABB*()
  ***while*** *network* is *online* *do*
    Connection $i$ from $x$ to server y is $\sigma_{ci(x)} \leftarrow (x, y)$
    $\mu_x \leftarrow false$
    Netflow class $N_c \leftarrow predictClassSVM\ \theta(N_{fx})$
    ***if*** $N_c$ *is attack*
      Sender-destination pair $(x, y) \leftarrow$ $getSenderDestPair(N_{fx}) : x \notin \{y_1 \cup \{y_\pi\}\}$
      ***if*** $y \neq y_n$
        $y' \leftarrow serverAfter(y)$
        New connections $j$ from $x$ is $\sigma_{cj(x)} \leftarrow (x, y')$
      ***else if*** $y = y_n$
        $\mu_x \leftarrow true$
      $switchPropagate(\mu_x)$

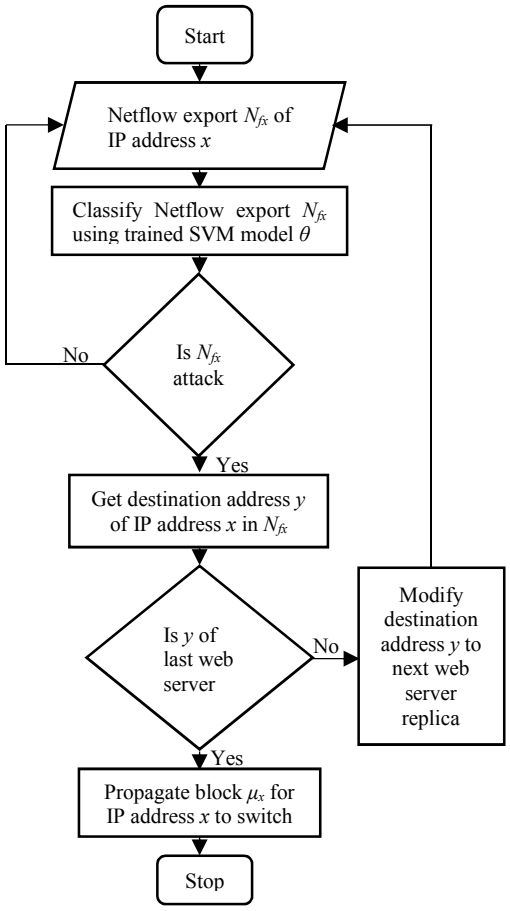


Fig. 1. Selective Adaptive Bubble Burst Algorithm

## V. RESULTS AND DISCUSSION OF THE DETECTION TECHNIQUE

Feature selection was executed first as the necessary features that aids in distinguishing an attack from a legitimate traffic is needed for effective choice of support vector machine parameters. When the features were obtained, the support vector machine parameter selection phase was launched and the appropriate regularization and gamma parameters were obtained. The selected features, regularization parameter, and gamma parameter were then used to perform the classification task using support vector machine.

### A. Feature Selection and Parameter Tuning Results

A total of 31 features were evaluated for distinguishing an attack from legitimate client traffic. The 31 features comprised of 27 netflow version 5 features and 4 other features of the time difference between the last and first packet of the flow in seconds, the number of packets per second, bytes per second, and bytes per packet. Thirteen features (13) were removed because they had the possibility of either causing overfitting of the classifier or had zero values in all tuples for the feature as shown in Table I. Out of the remaining 18 features, 11 features were selected as features that had a significant effect on distinguishing between an attack and legitimate client traffic as shown in Table II.

The tuned regularization and gamma value obtained are 8 and 0.798 respectively.

TABLE I: REDUCED FEATURES IN THE DATASET

| S/N | Feature Name | Description |
|---|---|---|
| 1 | Sys_Uptime | Current time in milliseconds since the export device booted |
| 2 | Unix_secs | Current count of seconds since 0000 UTC 1970 |
| 3 | Unix_nsecs | Residual nanoseconds since 0000 UTC 1970 |
| 4 | Flow_sequence | Counter of total flow sequence seen |
| 5 | Sampling Interval | Interval of netflow export sampling |
| 6 | Srcaddr | Source IP address |
| 7 | Dstaddr | Destination IP address |
| 8 | Nexthop | IP address of next-hop router |
| 9 | Tos | IP type of service |
| 10 | Src_as | Autonomous system number of the source |
| 11 | Dst_as | Autonomous system number of the destination |
| 12 | Src_mask | Source address prefix mask bits |
| 13 | Dst_mask | Destination address prefix mask bits |

TABLE II: SELECTED FEATURES IN THE DATASET

| S/N | Selected Features | Description |
|---|---|---|
| 1 | Count | Number of flows exported (1-30) |
| 2 | Input | Simple Network Management Protocol (SNMP) index of input interface |
| 3 | Output | SNMP index of output interface |
| 4 | DPkts | Packets in the flow |
| 5 | dOctets | Total number of layer 3 bytes in the packets of the flow |
| 6 | Last | SysUpTime at the time the last packet of the flow was received |
| 7 | Diff | Time difference in seconds between the last and first feature in the netflow version 5 feature set |
| 8 | Srcport | TCP/UDP source port number |
| 9 | Tcp_flags | Cumulative OR of TCP flags |
| 10 | Packets/second | Number of packets per second |
| 11 | Bytes/packet | Number of bytes per second |

### *B. Support Vector Machine Detection Result*

The selected features and the support vector machine parameters obtained were used to define the classification task. The dataset used had 56,891 tuples in total which contained 28,446 and 28,445 attack and legitimate client traffic tuples respectively. A 60:20:20 ratio for training, testing and validating the model obtained from the classification task was employed. Results obtained are described in table III.

TABLE III : PERFORMANCE OF THE SELECTED FEATURES

| Performance Metric | Percentage |
|---|---|
| Accuracy | 99.89% |
| AUC | 99.89% |
| TPR | 99.95% |
| FPR | 0.18% |
| FNR | 0.05% |

From the results obtained in Table III, it is established that detection accuracy is high. Also, the True Positive Rate (TPR) shows that most of the attack traffic were classified correctly. The False Positive Rate of 0.18% shows the percentage of legitimate traffic which was misclassified as attack

## VI. CONCLUSION

As examined, few research into detecting the three major slow HTTP DDoS attacks has been carried out. Furthermore, the absence of a publicly available slow HTTP DDoS dataset based on netflow version 5 necessitated the creation of a dataset by simulating an SDN network in GNS3. The results obtained by the genetic algorithm with support vector machine classifier indicates buttresses the use of machine learning for detecting anomalies.

In this paper, we conducted detection of slow HTTP DDoS attacks executed using genetic algorithm to select netflow version 5 features and tune the regularization and gamma parameters of the support vector machine classifier used. Detecting the slow HTTP DDoS attack effectively enables the mitigation mechanism to be triggered and applied to the traffic that caused the event. The algorithm for the selective adaptive bubble burst mitigation process was also proposed

Since accurate detection of attack ought to be followed by robust mitigation techniques, future research direction is to implement the selective adaptive bubble burst algorithm developed in the simulated network's controller. Furthermore, for the mitigation mechanism to be operational, the machine learning slow DDoS attack detection model generated is to be tested in the simulated network. Also, a machine learning framework that learns from the traffic classified in the live or simulated network is a possible area for future studies. This enables the detection mechanism to be updated based on the results it obtains after classifying live traffic and ensures that the detection mechanism remains relevant without a need to create an entirely new model at intervals.

## REFERENCES

[1] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "SOFTWARE-DEFINED NETWORKING SECURITY : PROS AND CONS," no. June, pp. 73–79, 2015.

[2] R. Swami, M. Dave, and V. Ranga, "Defending DDoS against Software Defined Networks using Entropy," in *Proceedings - 2019 4th International Conference on Internet of Things: Smart Innovation and Usages, IoT-SIU 2019*, pp. 1–5.

[3] E. Cambiaso, G. Papaleo, and M. Aiello, "Slowcomm: Design, development and performance evaluation of a new slow DoS attack," *J. Inf. Secur. Appl.*, vol. 35, pp. 23–31, 2017.

[4] G. A. Jaafar, S. M. Abdullah, and S. Ismail, "Review of Recent Detection Methods for HTTP DDoS Attack," *Journal of Computer Networks and Communications*, vol. 2019.

[5] N. Muraleedharan and B. Janet, "Behaviour analysis of HTTP based slow denial of service attack," in *Proceedings of the 2017 International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2017*, 2018, vol. 2018-Janua, pp. 1851–1856.

[6] E. Cambiaso, G. Papaleo, G. Chiola, and M. Aiello, "Slow DoS attacks: definition and categorisation," *Int. J. Trust Manag. Comput. Commun.*, vol. 1, no. 3/4, p. 300, 2013.

[7] S. Suroto, "A Review of Defense Against Slow HTTP Attack," *JOIV Int. J. Informatics Vis.*, vol. 1, no. 4, p. 127, 2017.
[8] A. Dhanapal and P. Nithyanandam, "The slow http distributed denial of service attack detection in cloud," *Scalable Comput.*, vol. 20, no. 2, pp. 285–298, 2019.
[9] S. Tayama and H. Tanaka, "Analysis of Slow Read DoS Attack," 2018.
[10] M. Idhammad, K. Afdel, and M. Belouch, "Detection System of HTTP DDoS Attacks in a Cloud Environment Based on Information Theoretic Entropy and Random Forest," *Secur. Commun. Networks*, vol. 2018, 2018.
[11] R. Swami, M. Dave, and V. Ranga, "Software-defined Networking-based DDoS Defense Mechanisms," *ACM Comput. Surv.*, vol. 52, no. 2, p. 36, 2019.
[12] M. Yeasir, M. Morshed, and M. Fakrul, "A Practical Approach and Mitigation Techniques on Application Layer DDoS Attack in Web Server," *Int. J. Comput. Appl.*, vol. 131, no. 1, pp. 13–20, 2015.
[13] N. Beigi-Mohammadi, C. Barna, M. Shtern, H. Khazaei, and M. Litoiu, "CAAMP: Completely automated DDoS attack mitigation platform in hybrid clouds," in *2016 12th International Conference on Network and Service Management, CNSM 2016 and Workshops, 3rd International Workshop on Management of SDN and NFV, ManSDN/NFV 2016, and International Workshop on Green ICT and Smart Networking, GISN 2016*, 2017, pp. 136–143.
[14] T. Lukaseder, L. Maile, B. Erb, and F. Kargl, "SDN-assisted network-based mitigation of slow DDoS attacks," in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 2018, vol. 255, pp. 102–121.
[15] L. Schehlmann and H. Baier, "COFFEE : a Concept based on OpenFlow to Filter and Erase Events of botnet activity at high-speed nodes," *GI-Jahrestagung*, pp. 2225–2239, 2013.
[16] S. S. Bhunia and M. Gurusamy, "Dynamic attack mitigation using SDN," in *2017 27th International Telecommunication Networks and Applications Conference, ITNAC*, vol. 2017-Janua, pp. 1–6.
[17] B. Yuan, D. Zou, H. Jin, S. Yu, and L. T. Yang, "HostWatcher: Protecting hosts in cloud data centers through software-defined networking," *Futur. Gener. Comput. Syst.*, 2017.
[18] I. E. Fonseca and V. Nigam, "Mitigating High-Rate Application Layer DDoS Attacks in Software Defined Networks," 2016.
[19] D. Ameyed, F. Jaafar, and J. Fattahi, "A slow read attack using cloud," in *Proceedings of the 2015 7th International Conference on Electronics, Computers and Artificial Intelligence, ECAI*, 2015, pp. SSS33–SSS38.
[20] D. Sattar, A. Matrawy, and O. Adeojo, "Adaptive Bubble Burst (ABB): Mitigating DDoS attacks in Software-Defined Networks," in *2016 17th International Telecommunications Network Strategy and Planning Symposium, Networks 2016 - Conference Proceedings*, 2016, pp. 50–55.
[21] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, "Software-defined networking (SDN): a survey," *Secur. Commun. Networks*, vol. 9, no. 18, pp. 5803–5833, 2016.
[22] D. J. Hamad, K. G. Yalda, and I. T. Okumus, "Getting traffic statistics from network devices in an SDN environment using OpenFlow," *Inf. Technol. Syst. 2015*, no. April, pp. 951–956, 2016.
[23] C. Kemp, C. Calvert, and T. M. Khoshgoftaar, "Utilizing netflow data to detect slow read attacks," in *Proceedings - 2018 IEEE 19th International Conference on Information Reuse and Integration for Data Science, IRI*, 2018, pp. 108–116.
[24] C. L. Calvert and T. M. Khoshgoftaar, "Impact of class distribution on the detection of slow HTTP DoS attacks using Big Data," *J. Big Data*, 2019.
[25] M. Siracusano, S. Shiaeles, and B. Ghita, "Detection of LDDoS Attacks Based on TCP Connection Parameters," in *Global Information Infrastructure and Networking Symposium*, 2018.
[26] M. Zolotukhin, T. Hamalainen, T. Kokkonen, and J. Siltanen, "Increasing web service availability by detecting application-layer DDoS attacks in encrypted traffic," in *2016 23rd International Conference on Telecommunications, ICT*, 2016.
[27] M. Shtern, R. Sandel, M. Litoiu, C. Bachalo, and V. Theodorou, "Towards mitigation of low and slow application DDoS attacks," in *Proceedings - 2014 IEEE International Conference on Cloud Engineering, IC2E*, 2014, no. Vm, pp. 604–609.
[28] B. Cusack and Z. Tian, "Detecting and tracing slow attacks on mobile phone user service," in *Proceedings of the 14th Australian Digital Forensics Conference, ADF*, 2016, no. December, pp. 4–10.
[29] N. Tripathi, N. Hubballi, and Y. Singh, "How Secure are Web Servers? An empirical study of Slow HTTP DoS attacks and detection," in *Proceedings - 2016 11th International Conference on Availability, Reliability and Security, ARES*, 2016, pp. 454–463.
[30] N. Tripathi and N. Hubballi, "Slow rate denial of service attacks against HTTP/2 and detection," *Comput. Secur.*, vol. 72, pp. 255–272, 2018.
[31] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling," *Comput. Networks*, vol. 121, pp. 25–36, 2017.