Predicting Attack Surface Effects on Attack Vectors in an Open Congested Network Transmission Session by Machine Learning

https://doi.org/10.3991/ijoe.v17i11.25025

Nahla Aljojo
University of Jeddah, Jeddah, Saudi Arabia
nmaljojo@uj.edu.sa

Abstract—This paper examined the impact of a network attack on a congested transmission session. The research is motivated by the fact that the previous research community has neglected to evaluate security issues related to network congestion environments, and has instead concentrated on resolving congestion issues only. At any point in time, attackers can take advantage of the congestion problem, exploit the attack surface, and inject attack vectors. In order to circumvent this issue, a machine learning algorithm is trained to correlate attack vectors from the attack surface in a network congestion signals environment with the value of decisions over time in order to maximise expected attack vectors from the attack surface. Experimental scenario that dwell on transmission rate overwhelming transmission session, resulting in a standing queue was used. The experiment produced a dataset in which a TCP transmission through bursting transmission were capture. The data was acquired using a variety of experimental scenarios. Nave Bayes, and K-Nearest Neighbours prediction analyses demonstrate strong prediction performance. As a result, this study re-establishes the association between attack surface and vectors with network attack prediction.

Keywords—attack surface, attack vectors, congestion, transmission session, machine learning

1 Introduction

Network transmission session defines the flow of data in connection-oriented communications among the connecting. Transmission Control Protocol (TCP), is dependable in a network transmission session because it employs feedback mechanisms to ensure that all data is delivered to the intended recipient. When compared to other protocols, it does not take into account the possibility of data delivery delays and may result in data being delivered slowly [1]. TCP continues to be the most widely used mode of communication. The issue of delay in transmission with TCP, is its weakness. While TCP's primary characteristics are its ability to transport data, it also provides services that are intended to improve reliability by dealing with data segments that have been lost during a transmission session as a result of congestion caused, most likely, by a large number of segments competing for limited network resources [2]. As a results it

calls for the implementation of congestion control measures. These services include dealing with loss data, minimising errors, managing a transmission session's failure. The fundamental challenges and issues in the design of a transport protocol are "mobility management", "bandwidth estimation", "packet loss estimation", “Quality of Service Support” among other things. Congestion control is applied at the transport layer in order to maintain the steady sending rate during periods of high-volume transmission. The congestion control mechanism detects packet losses and, as a result, reduces a percentage of the available congestion window. The common drawbacks og being into congested network environment is open-space for attack surfaces to exploit the network, and perform a Denial of Service Attack (DOS) or Distributed Denial of Service Attack (DDOS) [3].

Attack surfaces are the areas of a computer system that are being targeted by a cybercriminal. They are made up of all the assets that are exposed to an attacker and that can be exploited [4]. Attack Vectors are the specific method or pathway that a cybercriminal employs in order to conduct malicious activities against a targeted system or network [5].

Apart from DDoS or DDOS attacks, what other types of attacks are expected in a congestion situation? It was revealed that congestion can act as an implicit signal for decentralized implementation pulsating attacks, means that congestion itself is an attack surface [6]. This is a critical situation that necessitates assessment, because TCP is a connection-oriented protocol, it enables the control of data during transmission, which is necessary because an uncontrolled data flow can result in damage or loss. Attackers will redirect transmission flow until the congestion issue is resolved, taking advantage of this situation where the primary concern is to resolve data flow. As a result, once the congestion issue is resolved, the traffic direction will change unexpectedly [7]. This indicates the establishment of an attack surface. While the TCP connection is negotiating between the sending and receiving network nodes, specifically about the size of the data being transmitted, attacks are busy attempting to redirect the connection. They will eventually inject the attack vectors if they succeed. In a client-server architecture, the client informs the server of the maximum amount of data it is willing to accept from the server at one time. Similarly, the server must inform the client of the maximum amount of data it is willing to accept from the client at any given time, referred to as the server's receive window, which doubles as the client's send window [8]. This negotiation creates an opening for the attack surface.

Considering the nature of uncertainty within the transmission session associated with congestion as well as the fact that congestion is both an implicit signal for decentralised implementation attacks, and a legitimate cause of implementation attacks in and of itself. This paper examined the impacts of a network attack on a congested transmission session, and it considered how to deal with congestion issues associated with network attacks. To obtain training data from a network attack surface environment and to correlate the network attack impact with decision value changes over time, a machine learning algorithm was used.

2 Attacks associated with transmission on congested network

Apart from DOS or DDOS attacks, any other attacks associated with congestion or inaccessibility of service, are usually resolves by re-establishing paths to create more open paths necessary for transmission. That is why some studies reveals that all attacks associated to congestions are a different form of DOS or DDOS [6]. As a result, it could be concluded that attacks associated with transmission on congested are link-related TCP traffic. Pulsating attacks greatly reduced TCP traffic by continuous pulses at a network link [6]. The attack can take various forms, ranging from time-out-based pulsating attacks, to synchronous, asynchronous attacks, and traditional link flooding attacks [9]. Decentralized implementations of pulsating attacks that require a central coordinator can utilise congestion as an implicit signal for decentralisation. This creates an intermittent flooding which can reduce legitimate traffic and thus has the potential to be more harmful than simple brute-force flooding. For example, a series of carefully crafted periodic pulses on a network link can mislead TCP flows using the same link, causing legitimate flows to repeatedly enter the timeout state.

This study envisioned scenarios where the attacks described above can be predicted (see Figure 1). The interaction of transmission congestion associated to attack surface and vectors is conceptualized for the modelling of the prediction of attack surface effects on attack vectors in an open congested network transmission session. The justification of the claim lies with the vulnerability available in a TCP connection characterized with the attack associated with an attacker exploiting the attack surface. In order to gain access to an internal network in order to attack the network through the "hole" in the attack surface, it is possible that transmission session is unaware of this and are unaware of how the attack works, let alone what it allows and does not allow an attacker to accomplish.

Although the vulnerability to attack may be a critical issue in a network transmission session that is experiencing congestion, it is not the most pressing issue that requires extensive effort to resolve [10]. The conceptualization in Figure 2, established that TCP transmission, may continue in the event that the connection establishment in a 3-way handshake is maintained. A TCP split-handshake connection can be performed by an attacker in a situation where the TCP connection establishment is not monitored by TCP SYN Checking [11]. That is to say the ability to implement "TCP SYN Checking" will almost certainly result in an attack on the network. Thus, we have a special case in which an attack surface has been established [12]. The TCP SYN checking function, as a result, can protect the transmission from TCP state-based attacks [13].

In another situation, even if the TCP SYN checking was implemented, it would still be ineffective due to the fact that the connection establishment technique used in transmission was well known and that many defence security parameters will enforce the defences. The unfortunate reality is that there are additional connection establishments that are normally created when the intent is solely to attack a network infrastructure. The simultaneous-open handshake is the term used to describe this method of connection establishment [11]. When faced with this situation, the act of establishing the simultaneous-open handshake itself creates an attack surface. A simultaneous open transmission session will be established as a result, and both connections will send a SYN

packet to each other at approximately the same time, as shown in Figure 1. Then, as a response, both sides send each other acknowledgement packets (ACKs). This slightly different variant of the TCP handshake occurs infrequently in the real world; however, it is a perfectly legitimate way to initiate a TCP connection; however, it is most frequently used with the intent of committing an attack.

With the simultaneous-open handshake comes the TCP split-handshake attack, which is also known as a Snitch ACK attack in some circles [14]. The split-handshake, is a dual track that combines elements of the "normal three-way handshake" and the "simultaneous-open handshake" into a single gesture. Essentially, a client sends a SYN packet to a server with the intent of completing a standard three-way handshake. Instead of completing the three-way handshake initiated by the client, a malicious attacker begins by responding as if it were initiating a simultaneous-open connection, and then initiates its own three-way handshake in the opposite direction of the transmission session. In essence, even though the connection to the other side of the transmission session was initiated by the client, the flow direction of the connection is reversed as a result of this.

As presented in Figure 1, attack vectors are at malicious drive-in, at this point, it indicates that an attack surface has been established, and the attack vectors have been successfully evaded the connection. When this occurs, the malicious attacker has the opportunity to exploit the situation by reversing the logic direction of the connection that was originally established in his or her favour. That implies that there is a correlation between the network connection and the malicious attacker before the attack can even begin. When this attack is successful, the attacker does not even have complete control over the victim's connection; instead, the attacker has only reversed the logical direction of the initial connection.

This kind of attack could be harmful because it could result in the logical reversal of the direction of a perfectly legitimate connection that has already been established. This does not necessarily imply that the attacker can perform any new actions on the transmission connection, but it may cause confusion among the security services that are in charge of protecting the connection. If the malicious attacker uses a TCP split-handshake connection to launch the same attack, the network defence systems may become confused by the direction of the traffic and fail to scan the connection content for malicious activity. The malicious drive-through would now be successful, despite the fact that defensive security protection was in place. As a result, the TCP split-handshake attack may be used by malicious attackers to circumvent the security active services of a communication connection. Internal connections must initiate the connection in order for external attackers to be able to bypass network security policies. This prevents external attackers from circumventing any security policies.

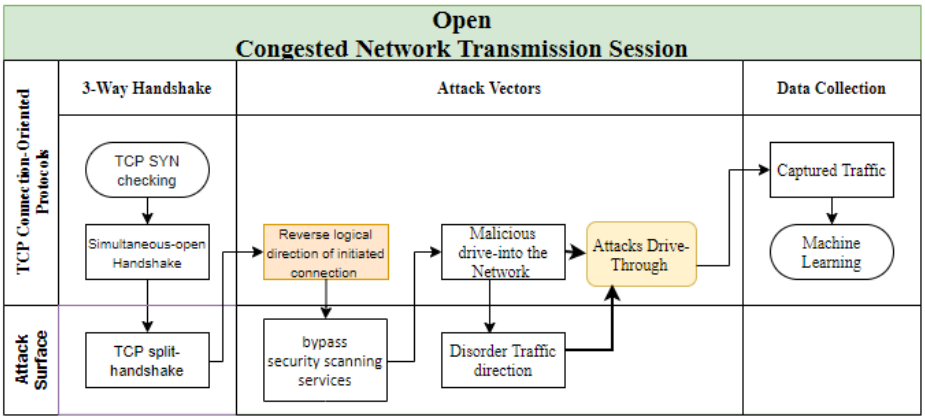

Fig. 1. The Interaction of Transmission Congestion Associated to Attack Surface and Vectors

3 Methodology and experimental analysis

This study opens several TCP connections scenario in which TCP transmission were captured by bursting transmission and overwhelming the transmission link. In a system running Kali Linux, a Wireshark was used and the count the numbers of instances of the SYN, SYN-ACK, ACK, http, and https with open TCP/UDP socket. Each incoming and outgoing TCP packet contains information about the network communication data, including elements such as (bytes-in-flight), connection syncs, data-text lines, duplicate acks, and flags. Open TCP/UDP socket is the attack vector, while the rest of the parameters are the attack surface as described in section 2. Even though, there are benchmark cybersecurity datasets for the evaluation of machine learning performance [15], this study generates its datasets considering that the situation here is based on congested network. Furthermore, the goal of prediction of the network security situation is to accurately predict the network security situation in real time. Data in a network security situation is random, ambiguous, and uncertain [16].

3.1 Machine prediction approach

Prediction problems in machine learning are tasks that consist of predicting the attributes of the next object based on the attributes of the object that was previously observed. Due to the high level of uncertainty in network security situation, prediction is difficult [16]. Common problem that finds real-world applications in a variety of network security areas is the prediction of the relationship between the attack surface and the attack vector. In a network congestion environment, a predictive model of the relationship between attack surface and attack vector can then be used to make predictions about previously unidentified security issues in the network. The problem of predicting the relationship between the attack surface and the attack vector critical conditions has a wide range of applications, and solving it has a large number of them. An attack surface and an attack vector prediction system is initially developed as a tool to solve a

security problem that does not perform well. This is when prediction of the relationship between the attack surface and the attack vector first appears. While considering a prediction model for critical situations in technical areas, performance is critical in this situation [17-19]. The current study made use of the prediction algorithm of machine learning and the dataset that was generated, which was divided into two groups: a training set and a test set. Using the training dataset, we can train the predefined techniques, and the test dataset allows us to evaluate the accuracy of the trained model that we have generated. In this study, the Nave Bayes, and K-Nearest Neighbors (K-NN), were used to make successful machine learning prediction predictions.

3.2 K-Nearest Neighbors (K-NN)

In numerous research projects [20-22], the K-NN model has been applied to network security, specifically intrusion detection. A non-parametric classifier, it operates under the assumption that "things that look alike must be alike." The Nearest Neighbor (NN) rule is an extension of the Nearest Neighbor (K-NN) rule, which was derived from the K-NN rule. Using a given vector space as a test space, it classifies sets of attributes by comparing each one with the entire class of its K nearest neighbours. So K is the parameter that can have several different values near the sample test, as demonstrated above. Due to its limitations, the K-NN algorithm is not always effective in producing good results, which is a disadvantage. Nonetheless, because of its simplicity of implementation and high performance, it is suitable for a wide range of problems. Given the fact that data in a network security environment is typically unpredictable, confusing, and uncertain, Rao and Swathi [20] use K-NN for the characteristics of denial-of-service (DoS) and probe attacks.

3.3 Naïve bayes algorithm

The Naive Bayes algorithm is one of the most widely used data mining algorithms. This method's efficiency is derived from the assumption of attribute independence; however, this assumption may be violated in many real-world data sets. There have been numerous efforts made to mitigate the assumption, with attribute selection being one of the most important approaches [23]. Although the Naive Bayes Algorithm is oversimplified and heavily reliant on prior knowledge, it is important for network security because it is fast and accurate. Through the use of many different features, such as network features, it tends to evaluate the likelihood of the expected outcome using the preceding probability model, which is then used to make the decision [24]. It is determined by using the Naive Bayes function whether a parameter or set of parameters is relevant or non-relevant to the objective function in question. As a result, the data will be classified by the model according to whether or not they are attack vectors that influence the attack surface. The attribute assumptions about a set of n attributes are revealed by this method, which is used for intrusion detection [25]. Most of the time, the predictions made by the naive Bayes classifier are correct. Currently, there are some

modified versions of naive bayes algorithm for network intrusion detection in the machine learning, a typical example is the one that applies artificial bee colony algorithm [26].

3.4 Evaluation metrics

The Evaluation is critical when estimating the performance of a machine learning algorithm. Typically, performance is measured using indicators such as precision and recall. Precision and recall are two different metrics that describe how well a prediction algorithm performs when rejecting a non-relevant class, and precision and recall are two different metrics that describe how well the algorithm finds all relevant classes. A binary label is used to differentiate between what happened in real life and what happened in the prediction when evaluating precision and recall, where *x* represents the correctness of the evaluation and *z* indicates relevance. This can be represented in the confusion matrix shown below:

		Assignment z	
		Predicted +	***Predicted -***
label x	Actual +	TP	FN
	Actual -	FP	TN

The positive and negative signs determine how the prediction reveals either the technique performance well or not; TP is "true positive" FP is "false positive", TN is "true negative", and FN is "false negative". Thus, precision and recall are derived from equations 1 and 2 respectively.

$$P = \frac{TP}{TP + FP} \tag{1}$$

$$R = \frac{TP}{TP + FN} \tag{2}$$

Where the technique do not find the dataset used in building the model appropriate it can then be concluded that TN+FP = 0, and when TP + FN = 0, it means there is no objects that is significant in the proposed the test set. This means that when evaluating the technique, errors are associated with false positives (the prediction of a non-relevant relationships) and false negatives (a relevant relationship is not found). For this reason, F measure called the F-score which is based on recall and precision is used for evaluating the prediction performance by equation 3.

$$F = \frac{2RP}{(R + P)} \tag{3}$$

Where R is the recall and P is the for precision

4 Presentation of the results

It is stated earlier that a variety of machine learning techniques are employed for network security prediction, all of which automate the process of identifying and constructing intellectual decisions based on data. This research employs a prediction model based on the network transmission session which becomes overloaded due to the amount of data that had reached a congestion. The captured dataset includes session data that was used to develop the model, and there was a correlation established, with some sets of data being conceptualised as attack surface and a category as the attack vector.

A procedure known as modelling was used to connect sets of input variables known as attack vectors and attack surfaces. Wireshark was used to capture data in several flow analyses. This solution set out two rules: The input variables could be utilised as groups of pairs of input in training data, and also input variables with a lower variance would increase the model's accuracy. In this way, two separate training and testing datasets were provided, each with their own partitioning of the dataset. To test the hypothesis, two analyses on the model were conducted, and two different partitioning schemes were used. In the first analysis, the partition dataset was not utilised, even though it was involved in the overall analysis. Labels apply to the Open TCP/UDP sockets dataset entirely. For prediction of sets, the dataset for the entire Naïve Bayes experiments was set at a threshold of 50% The values that were used for the K-NN experiment were set to 1, 2, and 3.

The performance analysis of the model for which the first partition of the dataset was used are presented in table 1 to 3. For the first model K-NN has been found to outperformed Naïve Bayes (see Table 1).

Table 1. The Performance Evaluation of the First Model

Partition	**RMSE**	**R2**	**MAE**	**RAE**
Naïve Bayes	1.0622	0.9146	0.9575	0.0061
K-NN	1.0591	0.9446	0.8532	0.0027

The Nave Bayes and KNN perform better than the first models in the second model, which is not similar to the first model. The second models divide training, validation, and testing into 80:10:10 ratios, whereas the first model divided training, validation, and testing into 70:15:15 ratios. As a result, it can be concluded that when the amount of training is increased, the model's performance is improved (see Table 2).

Table 2. The Performance Evaluation of the Second Model

Partition	**RMSE**	**R^2**	**MAE**	**RAE**
Naïve Bayes	1.1502	0.9398	0.9251	0.0092
K-NN	1.3271	0.9622	0.8485	0.0432

When comparing the last model to the first and second models, the performance of Nave Bayes and KNN models suffers. This final model divides training, validation, and

testing into three groups with the following ratios: 65:15:20. This also further demonstrates that when the number of training partitions is reduced, the performance evaluation will suffer (see Table 3).

Table 3. The Performance Evaluation of the Last Model

Partition	RMSE	R^2	MAE	RAE
Naïve Bayes	1.0034	0.9082	0.9575	0.0092
K-NN	1.0213	0.8337	0.8634	0.0416

The experimental trials are conducted to test text classification algorithms on test data. For classifying each dataset, the attributes of each dataset served as the classifiers. These trials' results are displayed in the figures in Figure 2 through 6. KNN performed the best of all three models in the test dataset as shown in the figures (Figures 2 and 3). In actual practise, perfect prediction rarely occurs. However, with the model and dataset, the accuracy of K-NN on all four performance metrics comes to 100% as shown in Figure 3. Figure 5 presents the final results for all the models, with the exception of a few (where it failed completely).

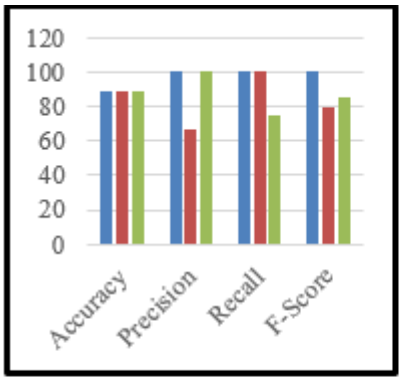

Figure 2. First Model Naïve Bayes Result

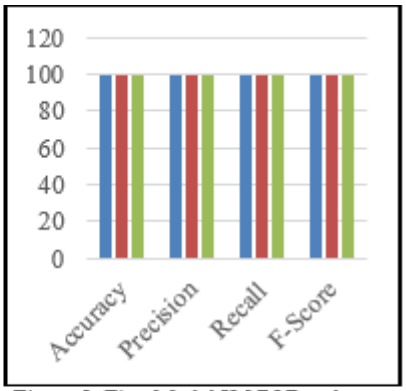

Figure 2. First Model K-NN Result

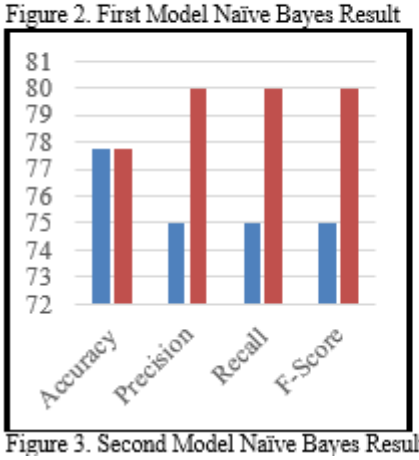

Figure 3. Second Model Naïve Bayes Result

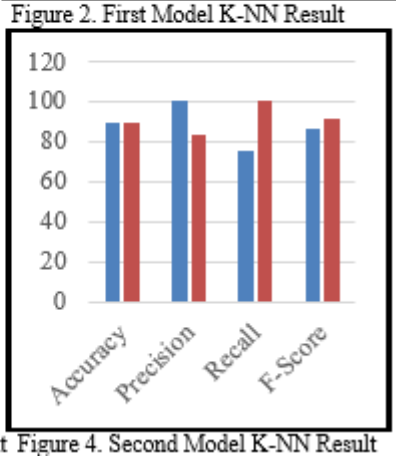

Figure 4. Second Model K-NN Result

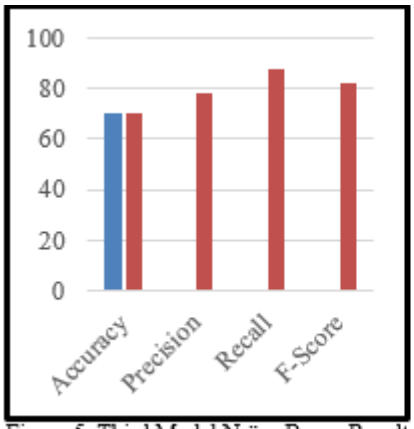

Figure 5. Third Model Naïve Bayes Result

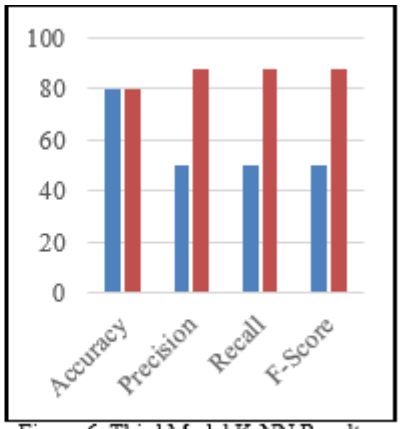

Figure 6. Third Model K-NN Result

This finding has demonstrated that using congestion as an implicit signal for decentralisation was can leads to attacks that required a central coordinator. The goal of connecting input variables using attack surface and attack vectors was to determine their relationships. In all the analyses carried out R^2 values are very high, it should be recognised that these analyses used dataset from several flow analyses captured in Wireshark within these flows. The study also recognised that it is an urgent matter to establish a mechanism for identifying attacks on TCP connection-oriented congestion because it provides the ability to control data as it is being sent, and yet, if left unchecked, may cause data loss that would disrupt the connection. Attackers will divert the data flow in order to resolve the congestion problem but attack the network at the same time. Because attacks may be observed while the TCP connection negotiates details, specifically the size of the data being transmitted.

5 Conclusion

This paper acknowledges that attackers can use congestion as a mechanism to gain access to an application's resources and then utilise that attack surface to spread attack vectors. Attack surface refers to everything that provides services to a computer system, such as a network, an operating system, and various services that are executed. Attack surface includes anything on the network, such as the operating system, various services, and even applications. Based on the Nave Bayes and K-Nearest Neighbors predictions, it is concluded that in a congestion environment of TCP transmission session, the attack surfaces are highly influencing attack vector. The performance of the algorithm used for this these experiment have demonstrated outstanding performance. It can be seen from this study that by uncovering attack surfaces, we can restore the link between attack vectors and network attack prediction. When speaking of TCP, the data transport feature is most important, However, it also exploits attack surface, despite providing reliable communication services.

6 Acknowledgment

This work was funded by the University of Jeddah, Saudi Arabia, under grant No. (UJ-20-017-SAI). The author, therefore, acknowledge with thanks the University Technical and financial support.

7 Reference

[1] Abubakar, A., & Oo, K. H. (2018). Window Size and Round-Trip-Time in a Network Transmission Session. IEEE 2018 International Conference on Information and Communication Technology for the Muslim World (ICT4M) pp: 162-166, July, Kuala Lumpur, Malaysia. https://doi.org/10.1109/ict4m.2018.00038

[2] Gür, G., Bahtiyar, Ş., & Alagöz, F. (2015). Security analysis of computer networks: Key concepts and methodologies. Model. Simul. Eng. 861-898. https://doi.org/10.1016/B978-0-12-800887-4.00030-4

[3] Al-Naeem, M., Rahman, M. A., Ibrahim, A. A. B., & Rahman, M. M. H. (2020). AI-based techniques for DDoS attack detection in WSN: A systematic literature review. J. Comput. Sci, 16(6), 848-855. https://doi.org/10.3844/jcssp.2020.848.855

[4] Theisen, C., Munaiah, N., Al-Zyoud, M., Carver, J. C., Meneely, A., & Williams, L. (2018). Attack surface definitions: A systematic literature review. Inf Softw Technol, 104, 94-103. https://doi.org/10.1016/j.infsof.2018.07.008

[5] Hou, X., Breier, J., Jap, D., Ma, L., Bhasin, S., & Liu, Y. (2021). Physical security of deep learning on edge devices: Comprehensive evaluation of fault injection attack vectors. Microelectron Reliab, 120, 114116. https://doi.org/10.1016/j.microrel.2021.114116

[6] Ke, Y. M., Chen, C. W., Hsiao, H. C., Perrig, A., & Sekar, V. (2016). Cicadas: Congesting the internet with coordinated and decentralized pulsating attacks. In Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security pp: 699-710, May, Xi'an, China. https://doi.org/10.1145/2897845.2897866

[7] Singh, J., & Behal, S. (2020). Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions. Comput. Sci. Rev, 37, 100279. https://doi.org/10.1016/j.cosrev.2020.100279

[8] Glaroudis, D., Iossifides, A., & Chatzimisios, P. (2020). Survey, comparison and research challenges of IoT application protocols for smart farming. Comput. Netw, 168, 107037. https://doi.org/10.1016/j.comnet.2019.107037

[9] Kaur, G., Saxena, V., & Gupta, J. P. (2020). Detection of TCP targeted high bandwidth attacks using self-similarity. J. King Saud Univ. - Comput. Inf. Sci, 32(1), 35-49. https://doi.org/10.1016/j.jksuci.2017.05.004

[10] Rafique, W., Qi, L., Yaqoob, I., Imran, M., Rasool, R. U., & Dou, W. (2020). Complementing IoT services through software defined networking and edge computing: A comprehensive survey. IEEE Commun. Surv. Tutor. 22(3), 1761-1804. https://doi.org/10.1109/comst.2020.2997475

[11] Beardsley, T., & Qian, J. (2010). The TCP Split Handshake: Practical Effects on Modern Network Equipment. Netw. Protoc. Algorithms, 2(1), 197-217. https://doi.org/10.5296/npa.v2i1.285

[12] Nagai, R., Kurihara, W., Higuchi, S., & Hirotsu, T. (2018). Design and implementation of an openflow-based tcp syn flood mitigation. IEEE 2018 6th International Conference on

Mobile Cloud Computing, Services, and Engineering (MobileCloud) pp: 37-42, March, Bamberg, Germany, https://doi.org/10.1109/mobilecloud.2018.00014

[13] Ravi, N., Shalinie, S. M., Lal, C., & Conti, M. (2020). AEGIS: Detection and mitigation of TCP SYN flood on SDN controller. IEEE Trans. Netw. Serv. Manag,18(1), 745-759. https://doi.org/10.1109/tnsm.2020.3037124

Bock, K., Hughey, G., Merino, L. H., Arya, T., Liscinsky, D., Pogosian, R., & Levin, D. (2020). Come as You Are: Helping Unmodified Clients Bypass Censorship with Server-side Evasion. Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication pp. 586-598, July, Virtual Event USA. https://doi.org/10.1145/3387514.3405889

[15] Abubakar, A. I., Chiroma, H., Muaz, S. A., & Ila, L. B. (2015). A review of the advances in cyber security benchmark datasets for evaluating data-driven based intrusion detection systems. Procedia Computer Science, 62, 221-227. https://doi.org/10.1016/j.procs.2015.08.443

[16] Liu, D. (2020). Prediction of network security based on DS evidence theory. ETRI Journal, 42(5), 799-804. https://doi.org/10.4218/etrij.2019-0147

[17] Dokic, S. B., & Rajakovic, N. L. (2019). Security modelling of integrated gas and electrical power systems by analyzing critical situations and potentials for performance optimization. Energy, 184, 141-150. https://doi.org/10.1016/j.energy.2018.04.165

[18] Najeeb, S. M. M., Al-Nima, R. R. O., & Al-Dabag, M. L. (2021). Reinforced Deep Learning for Verifying Finger Veins. Int. j. online biomed. eng, 17(7). https://doi.org/10.3991/ijoe.v17i07.24655

[19] Ayad, H., Ghindawi, I., & Kadhm, M. (2020). Lung Segmentation Using Proposed Deep Learning Architecture. Int. j. online biomed. eng, 16, (15), 141-147, https://doi.org/10.3991/ijoe.v16i15.17115

[20] Rao, B. B., & Swathi, K. (2017). Fast kNN classifiers for network intrusion detection system. Indian J Sci Technol. 10(14), 1-10. https://doi.org/10.17485/ijst/2017/v10i14/93690

[21] Aburomman, A. A., & Reaz, M. B. I. (2016). A novel SVM-kNN-PSO ensemble method for intrusion detection system. Appl. Soft Comput., 38, 360-372. https://doi.org/10.1016/j.asoc.2015.10.011

[22] Li, L., Yu, Y., Bai, S., Hou, Y., & Chen, X. (2017). An Effective Two-Step Intrusion Detection Approach Based on Binary Classification and $ k $-NN. IEEE Access, 6, 12060-12073. https://doi.org/10.1109/access.2017.2787719

[23] Chen, S., Webb, G. I., Liu, L., & Ma, X. (2020). A novel selective naïve Bayes algorithm. Knowl Based Syst, 192, 105361. https://doi.org/10.1016/j.knosys.2019.105361

[24] Pinitkan, Suriya, and Nawaporn Wisitpongphan (2020). "Abnormal Activity Detection and Notification Platform for Real-Time Ad Hoc Network." Int. j. online biomed. Eng, 16, (15), 45-63. https://doi.org/10.3991/ijoe.v16i15.16065

[25] Yang, J., Ye, Z., Yan, L., Gu, W., & Wang, R. (2018). Modified naive bayes algorithm for network intrusion detection based on artificial bee colony algorithm. IEEE 2018 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS) pp: 35-40, September, Cracow, Poland. https://doi.org/10.1109/idaacs-sws.2018.8525758

[26] Rusland, N. F., Wahid, N., Kasim, S., & Hafit, H. (2017). Analysis of Naïve Bayes algorithm for email spam filtering across multiple datasets. IOP conference series: materials science and engineering, 226, (1), p: 012091, August, Cracow, Poland. https://doi.org/10.1088/1757-899x/226/1/012091

8 Author

Nahla Aljojo obtained her PhD in Computing at Portsmouth University. She is currently working as Associate Professor at College of Computer Science and Engineering, Information system and information Technology Department, University of Jeddah, Jeddah, Saudi Arabia. Her research interests include: adaptively in web-based educational systems, e-Business, leadership's studies, information security and data integrity, e-Learning, education, machine learning, health informatics, environment and ecology, and logistics and supply chain management. Her contributions have been published in prestigious peer-reviewed journals.

Article submitted 2021-06-23. Resubmitted 2021-08-05. Final acceptance 2021-08-06. Final version published as submitted by the author.