IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 16, 2017, accepted March 19, 2017, date of publication April 6, 2017, date of current version May 17, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2688460

An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment

AQEEL SAHI[1,2], DAVID LAI[2], YAN LI[2], (Member, IEEE), AND MOHAMMED DIYKH[1,2]

[1]Thi-Qar University, Nasiriya 64001, Iraq
[2]School of Agricultural, Computational and Environmental Sciences, University of Southern Queensland, Toowoomba, QLD 4350, Australia

Corresponding author: A. Sahi (akeel_sahy@yahoo.co.uk)

ABSTRACT Although the number of cloud projects has dramatically increased over the last few years, ensuring the availability and security of project data, services, and resources is still a crucial and challenging research issue. Distributed denial of service (DDoS) attacks are the second most prevalent cybercrime attacks after information theft. DDoS TCP flood attacks can exhaust the cloud's resources, consume most of its bandwidth, and damage an entire cloud project within a short period of time. The timely detection and prevention of such attacks in cloud projects are therefore vital, especially for eHealth clouds. In this paper, we present a new classifier system for detecting and preventing DDoS TCP flood attacks (CS_DDoS) in public clouds. The proposed CS_DDoS system offers a solution to securing stored records by classifying the incoming packets and making a decision based on the classification results. During the detection phase, the CS_DDOS identifies and determines whether a packet is normal or originates from an attacker. During the prevention phase, packets, which are classified as malicious, will be denied to access the cloud service and the source IP will be blacklisted. The performance of the CS_DDoS system is compared using the different classifiers of the least squares support vector machine (LS-SVM), naïve Bayes, K-nearest, and multilayer perceptron. The results show that CS_DDoS yields the best performance when the LS-SVM classifier is adopted. It can detect DDoS TCP flood attacks with about 97% accuracy and with a Kappa coefficient of 0.89 when under attack from a single source, and 94% accuracy with a Kappa coefficient of 0.9 when under attack from multiple attackers. Finally, the results are discussed in terms of accuracy and time complexity, and validated using a K-fold cross-validation model.

INDEX TERMS Classification, cloud computing, DDoS attacks, LS-SVM.

I. INTRODUCTION

Distributed denial of service (DDoS) TCP flood attacks are DoS attacks in which attackers flood a victim machine with packets in order to exhaust its resources or consume bandwidth [1]. As the attack may be distributed over multiple machines, it will be very hard to differentiate authentic users from attackers. In fact, a DDoS flood attack is not only a widespread attack; it is the second most common cybercrime attack to cause financial losses [2] according to the United States Federal Bureau of Investigation (FBI).

The use of cloud computing is quickly increasing in many sectors, and especially in the health sector, as a result of its vital features, such as availability and on-demand services [3]. Most people think of cloud computing as virtual network which can offer flexible and accessible on-demand services [4]. However, the author in [5] pointed out that cloud computing involves much more than this, which has led researchers to re-consider its security more seriously. In addition, as mentioned in an electronic cybercrime study published by KPMG in collaboration with eCrime Congress in 2009, most of the cloud's virtual clients are under threat, and these threats increase as time passes [6].

There are many procedures [7] which can be adopted to mitigate the DDoS flood attacks, such as classifications [8], [9], encryption techniques [10]–[12]. As DDoS flood attacks can be implemented in many forms, the form of these attacks cannot be foreseen. Therefore, our new proposed classifier system for the detection and prevention of DDoS TCP flood attacks (CS_DDoS) is classification based, and can identify these attacks data regardless of the form in which they arrive at the cloud system. Classification can be defined as a common procedure for classifying, distinguishing and

differentiating multiple objects. Different classifiers, such as least squares support vector machine (LS-SVM), naïve Bayes, K-nearest and multilayer perceptron [13], [14] are used in this study to perform the classification process.

This paper is organized as follows: in Section 2, we review related work, and Section 3 introduces the simulation platform, with and without DDoS TCP flood attacks. Section 4 presents our proposed CS_DDoS system, and its performance is evaluated and validated in Section 5. Finally, we conclude this study and discuss future work in Section 6.

II. RELATED WORK

Many detection and prevention methods for mitigating DDoS flood attacks have been reported in the last few years [15].

The rank correlation-based detection (RCD) scheme was proposed by Wei et al. in [16]. The authors of the RCD claimed that their scheme could distinguish whether the incoming requests were from genuine users or from attackers. In [17], the ALPi algorithm was introduced, which decreased difficulties in packet flows and improved functionality by extending the concept of packet scoring. The ALPi therefore raises the detection accuracy percentage and attack recognition. Another DDoS attack prevention architecture, known as secure overlay services (SOS), was presented in [18]. The SOS architecture is a combination of three parts: secure overlay tunneling, routing via consistent hashing, and filtering. The authors claimed that the SOS can successfully decrease the probability of these attacks using filtering close to the secure edge and randomness close to the front edge.

Moreover, Wang and Reiter proposed the web referral architecture for privileged service (WRAPS) [19]. The WRAPS adopted the structure of a web graph to resist DDoS flood attacks, and requires authentic users to be authenticated using a referral hyperlink from a trusted site. Another approach was introduced to detect application DoS attacks on backend servers called the group testing-based approach [20]. The authors extended the existing group testing approach by reallocating users' requests to several servers. Markov Chain probability theory was adopted by Salah et al. when proposing an analytical queuing approach which examines the performance of firewalls under DDoS attacks [21].

In addition, Dou et al. [22] presented a confidence-based filtering (CBF) scheme for cloud projects. In the CBF, packets of information from authentic users is gathered during non-attack periods to extract features, which can generate an information profile of these non-attack periods. With this profile, the CBF scheme will be endorsed using a packet-scoring calculation during attacks to make a decision on whether to remove these packets or not. Another approach to detecting flood attacks, the fast lightweight detection approach, was presented by Yu et al. [23]. This approach utilized SNMP-MIB (simple network management protocol-management information base) statistical data as an alternative to raw data, as well as a SVM classifier for attack classification. Lee et al. [24] introduced a practical DDoS detection scheme based on DDoS architecture. In this scheme, they selected variables based on particular features that were extracted from a DDoS architecture. A cloud trace back (CTB) method was proposed in [25]. The authors of the CTB claimed that their method could identify the sources of the attacks. They also proposed a cloud Protector (CP), which made use of a back-propagation classifier in order to detect such attacks.

Furthermore, a new framework was presented by Lu et al. in [26]. This framework was able to effectively identify compromised packets. It analyzed these packets at the router end using a perimeter-based DDoS prevention system. Wang et al. introduced a graphics-based DDoS attack prevention and detection scheme, which was able to work with the data shift issue [7]. This scheme works by prevention, using network monitoring and a precise response with an elastic control structure. In [27], an adaptive selective verification (ASV) system was proposed. The ASV does not rely on network assumptions, and utilizes bandwidth efficiently. Another approach was presented based on five features (average number of packets per flow, percentage of correlative flow, one-direction generating speed, ports generating speed, and percentage of abnormal packets) combined with a Bloom filter [28]. In this approach, only users on the whitelist are allowed to reach their destinations; this whitelist is generated to include legitimate users only. However, this approach was implemented on the switches side (i.e. in hardware), which makes any future amendments or updates challenging [29].

While many mechanisms have been proposed to detect and prevent DDoS flood attacks, most of these do not provide high accuracy and are not efficient or fast detection and prevention techniques [30]. Furthermore, many of the DDoS attack protection mechanisms described here face scalability issues due to the fact that networks are becoming larger and faster; in addition, industrial deployment needs to be considered [17].

Therefore, cloud computing needs an efficient DDoS mitigation approach that can offer fast and accurate detection while remaining scalable. The proposed CS_DDoS was designed with all of these factors in mind.

III. DDoS TCP FLOOD ATTACKS

DDoS attacks can be established in two different ways: either directly and/or indirectly [31], [32]. Direct attacks target a weakness in the system of the victim machines and damage the machines directly. On the other hand, indirect attacks do not target victim machines directly; they prey on other elements with which the victim machines are associated and hinder their work [33]. In the following discussion, the TCP flood attack is used; this is an indirect attack, as it consumes most of the network's resources, meaning that they are not readily available to other users.

A TCP flood attack was carried out using software on a virtual cloud network; Wireshark Network Analyzer 2.0.0 [34] was used to capture and analyze traffic both before and during the attack.

A. BEFORE THE ATTACK

The network was simulated as shown in Figure 1.

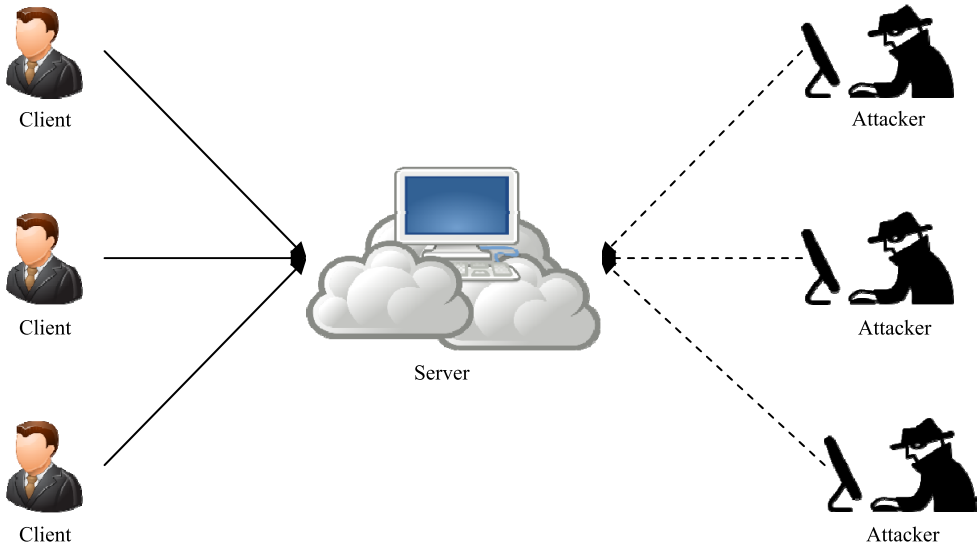


FIGURE 1. Test network architecture.

Firstly, using TCP Ping, we sent 50 TCP test probes (pings) to a server (server machine 10.25.129.5:80). The reply took 1.3 ms on average, as shown below:

```
Ping statistics for 10.25.129.5:80
  50 probes sent.
Approximate trip times in milliseconds:
  Minimum = 0.25 ms, Maximum = 26.065 ms,
  Average = 1.323 ms
```

The TCP protocol uses several flags to manage the state of a connection in the packet header [35]. We focused on two of these, which are used in establishing TCP connections:

- SYN (Synchronize) which represents the initiation of a connection; and
- ACK (Acknowledge) which represents data received.

We monitored the traffic of the 50 probes at the server machine using Wireshark, by capturing the packets that were associated with the server using the filter "ip.addr = = 10.25.129.5". As the traffic was normal, the server machine replied to all requested packets according to the TCP protocol, as shown in Figure 2 (a and b).

In addition, the I/O graph was stable. All packets were answered and almost no TCP errors occurred. Note that the number of requesting packets was approximately less than 10 per second, as shown in Figure 3.

B. DURING THE ATTACK

An attack was launched using a software program which performed a DDoS TCP flood attack on a particular server. Once the DDoS TCP flood attack commenced on the victim machine in the cloud, the arriving packets were much more numerous than the server could handle. Consequently, the server could not respond to all the requesting packets from either normal users or the attackers. Note that 10.25.129.5 was the IP address of the victim server and 10.31.133.235 was the IP address of the attacker. The first request packet from the attacker was successful, as it was treated like a normal requesting packet. The subsequent ones were not successful, as the server was too busy and could not respond. A screen shot of the packet capture is shown in Figure 4 (a and b).

Finally, we sent 50 TCP test probes within a few seconds to the victim machine during the attack period to test the connection. The reply time was 9.6 ms on average, which differs considerably from the first test as shown below:

```
Ping statistics for 10.25.129.5:80
  50 probes sent.
Approximate trip times in milliseconds:
  Minimum = 0.181 ms, Maximum = 152.341 ms,
Average = 9.586 ms
```

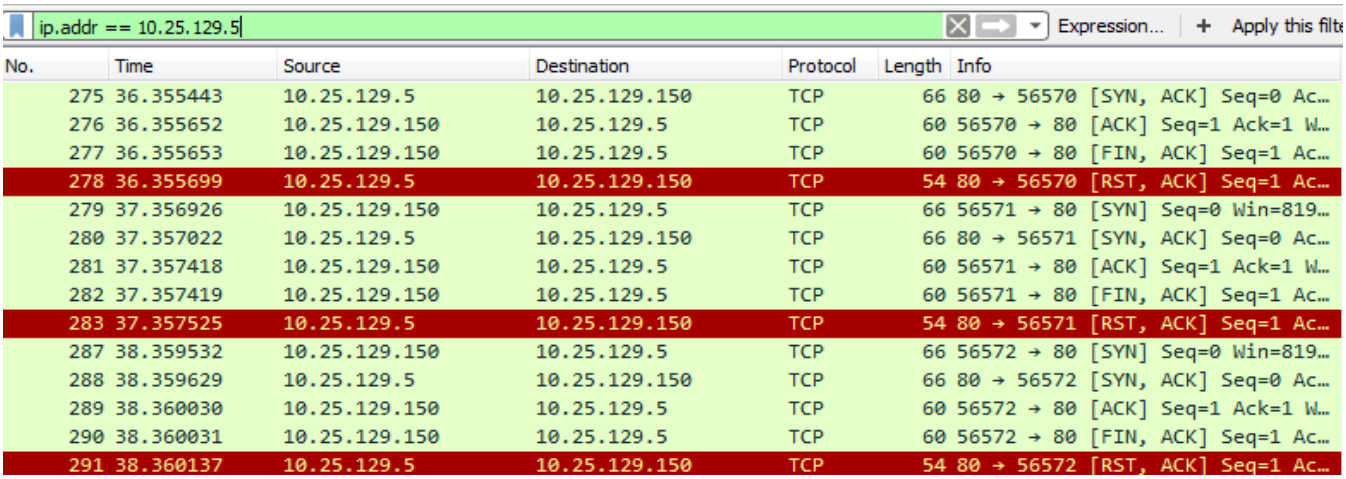
To sum up, the DDoS TCP flood attack can affect the cloud server's performance within a short time, slowing down the response, and can even stop the service completely. TCP errors will also be increased, as shown in Figure 5. Therefore, an efficient and effective detection and prevention technique is required.

IV. THE PROPOSED CS_DDoS SYSTEM

In this section we present the proposed CS_DDoS system, which can prevent DDoS TCP flood attacks. Firstly, it was assumed that the IP addresses of the attackers are not spoofed. Examples of how to prevent IP spoofing can be found in [36]. Our proposed system includes two sub-systems: the detection sub-system and prevention sub-system, as shown in Figure 6.

A. DETECTION PHASE

During the detection phase, the detection sub-system collects the incoming packets within a time frame, for example 60 seconds. The collected packets are subjected to a blacklist check to test whether their sources are blacklisted as attackers of the cloud system. If the packet source is listed in the attacker blacklist, the detection system will send the packets directly to the prevention sub-system without further processing.



ip.addr == 10.25.129.5 — Expression... — Apply this filt

No.	Time	Source	Destination	Protocol	Length	Info
275	36.355443	10.25.129.5	10.25.129.150	TCP	66	80 → 56570 [SYN, ACK] Seq=0 Ac…
276	36.355652	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [ACK] Seq=1 Ack=1 W…
277	36.355653	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [FIN, ACK] Seq=1 Ac…
278	36.355699	10.25.129.5	10.25.129.150	TCP	54	80 → 56570 [RST, ACK] Seq=1 Ac…
279	37.356926	10.25.129.150	10.25.129.5	TCP	66	56571 → 80 [SYN] Seq=0 Win=819…
280	37.357022	10.25.129.5	10.25.129.150	TCP	66	80 → 56571 [SYN, ACK] Seq=0 Ac…
281	37.357418	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [ACK] Seq=1 Ack=1 W…
282	37.357419	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [FIN, ACK] Seq=1 Ac…
283	37.357525	10.25.129.5	10.25.129.150	TCP	54	80 → 56571 [RST, ACK] Seq=1 Ac…
287	38.359532	10.25.129.150	10.25.129.5	TCP	66	56572 → 80 [SYN] Seq=0 Win=819…
288	38.359629	10.25.129.5	10.25.129.150	TCP	66	80 → 56572 [SYN, ACK] Seq=0 Ac…
289	38.360030	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [ACK] Seq=1 Ack=1 W…
290	38.360031	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [FIN, ACK] Seq=1 Ac…
291	38.360137	10.25.129.5	10.25.129.150	TCP	54	80 → 56572 [RST, ACK] Seq=1 Ac…

(a)

(b)

FIGURE 2. Captured packets and TCP flags (normal). (a) Captured packets. (b) TCP flags.

If the packet source is not blacklisted, the incoming packet will be passed to the classifier to decide whether the packets are normal (originating from a client) or abnormal (originating from an attacker). A packet is considered to be an attacking one if the source requests connections to the same destination more frequently than an assumed threshold. The threshold can be manually adjusted by the system administrator to cater for the varying requirements of a particular network. If a packet is considered to be normal, the detection system will send it to its destination (the cloud service provider). Otherwise, the detection sub-system will send the packet to the prevention sub-system.

Four different classifiers are used in the detection sub-system for the classification operation. The classifiers used are explained and evaluated in Section 5.

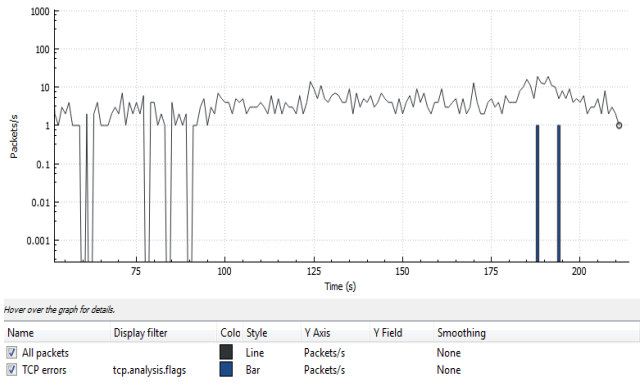
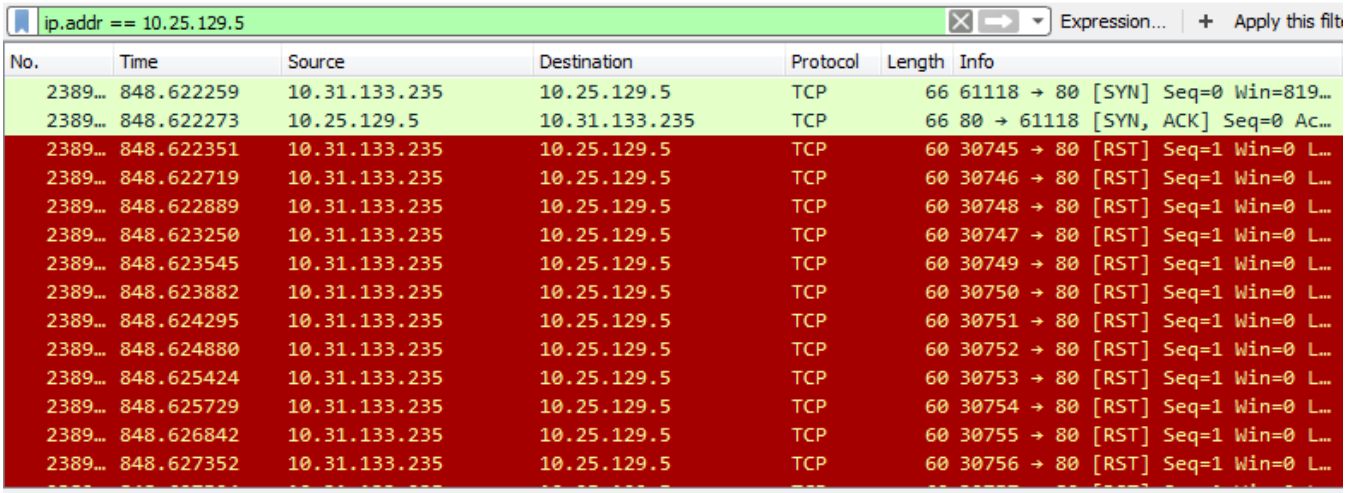


FIGURE 3. The I/O graph (no TCP errors).

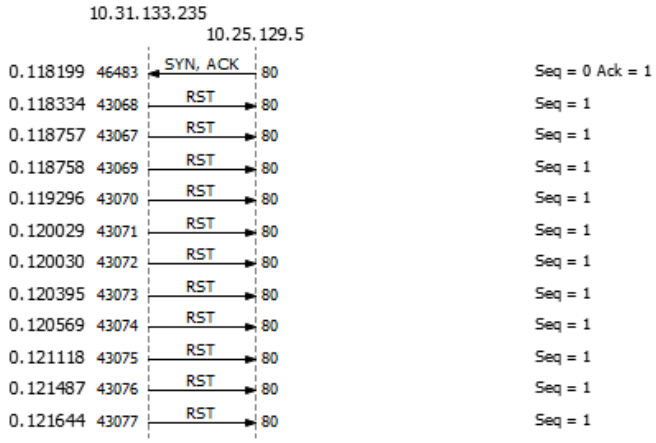
B. PREVENTION PHASE

When the packets reach the prevention system, they are considered to be attacking packets by the detection sub-system. The prevention sub-system first alerts the system administrator of the attacks. Then, the prevention sub-system will add the attacking source address to the attacker blacklist used by the detection sub-system, if it is not already on the list. Finally, the attacking packet will be dropped. The overall architecture of the CS_DDoS system is shown in Figure 6.

Algorithm 1 is used to determine whether these packets are normal or abnormal by counting the number of requests for a connection from an IP address and checking whether it



(a)



(b)

FIGURE 4. Captured packets and TCP flags (abnormal). (a) Captured packets. (b) TCP flags.

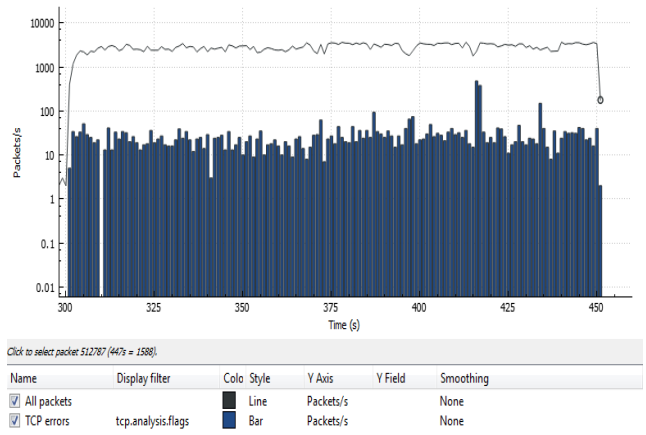


FIGURE 5. The I/O graph (with TCP errors).

exceeds a predefined threshold within a certain time frame. This algorithm is applied to the training data used for each classifier. As a result, each classifier used will predict the behavior of the attackers according to Algorithm 1.

Algorithm 1 Pre-Processing

```
Load data
For I=1: n
    P=data (I, 2)
    P2= (I, 1)
    For J=1: n
        N=find (data (J, 1) ==P2) & (data (J, 2) ==P)
        If N>=K
            New_data (I, 1) =data (I, 1)
            New_data (I, 2) =−1
        Else
            New_data (I, 1) =data (I, 1)
            New_data (I, 2) =1
End
```

where:
n is the number of packets
P is the destination IP address

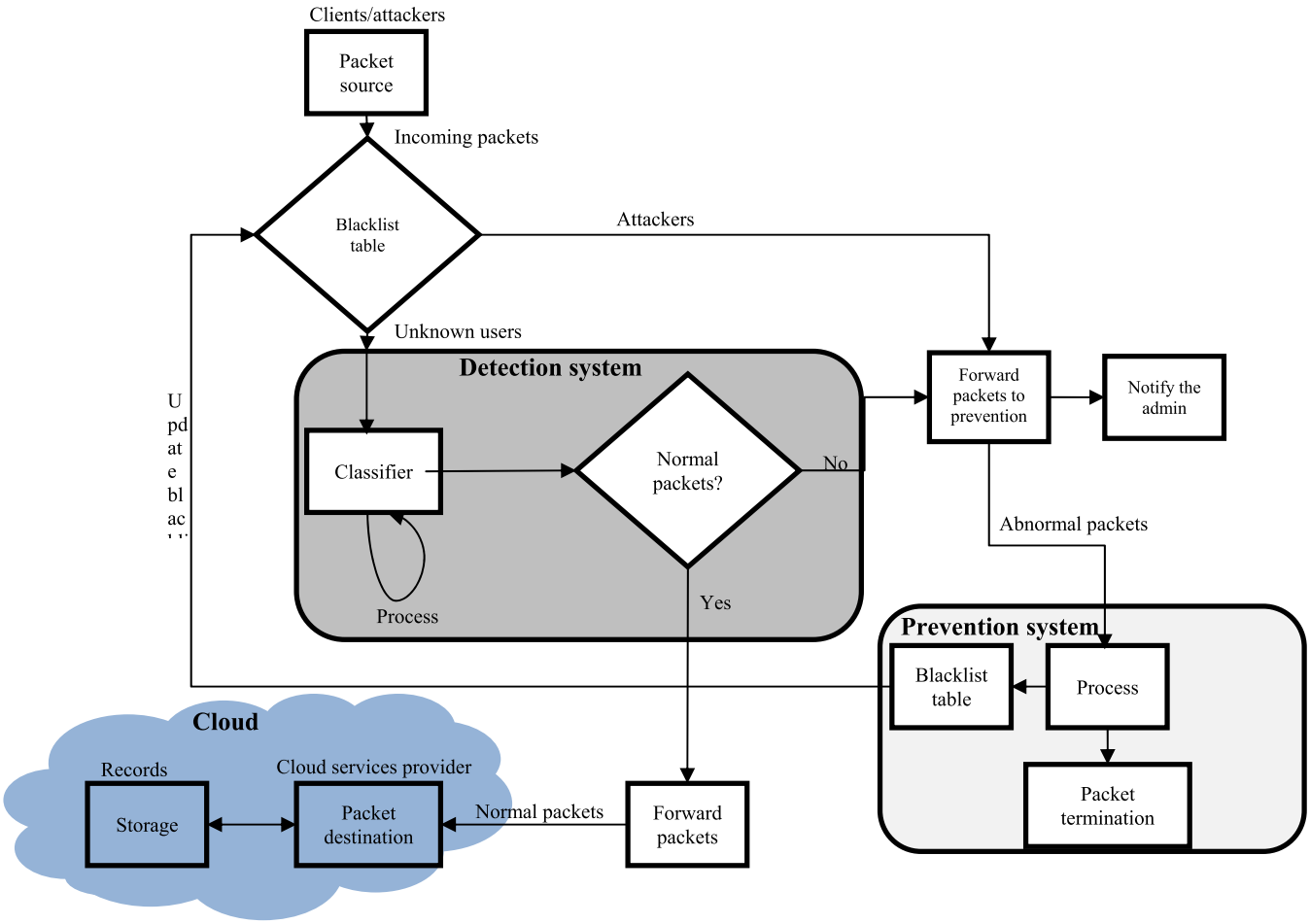


FIGURE 6. The overall architecture of the proposed CS_DDoS system.

P2 is the source IP address

N is the number of packets from the same source to the same destination within 60 seconds

K is the threshold for a packet to be considered an attacking packet

−1 indicates abnormal packets (blacklist array)

1 indicates normal packets

New_data () is a new entry list with tag "1" or "−1"

The proposed CS_DDoS system can be implemented in three possible scenarios. The first scenario is a normal service request packet. The requested service will be delivered as usual. The next scenario is when the source IP address is not blacklisted but the number of service requesting packets exceeds a predefined threshold within a certain time frame. The packet in this scenario will be considered a DDoS attack packet. The source address will be blacklisted and the packet will be dropped. The last scenario is when the source address of a packet is blacklisted and the packet is dropped without any further processing.

The three scenarios are illustrated using Quick Sequence Diagram Editor 4.2 [37]. The code used is shown in Table 11. The resulting sequence diagrams are shown in Figure 7 (a, b and c).

In case of flash crowd scenario, all packets must wait in a queue to be served sequentially.

The proposed CS_DDoS system can be used in any type of cloud, such as eHealth clouds, to ensure the security and availability of health records against DDoS TCP flood attacks.

V. EXPERIMENTAL RESULTS

A. CLASSIFICATION ALGORITHMS

In this section, we briefly explain the four commonly used classification algorithms used in our experiments. The classification algorithms are as follows:

1) LS-SVM

The LS-SVM is a powerful classifier in the field of pattern recognition for the detection of abnormalities from signals, images and time series signals. The LS-SVM is an efficient method of classifying two different sets of observations into their relevant classes. It is capable of handling high

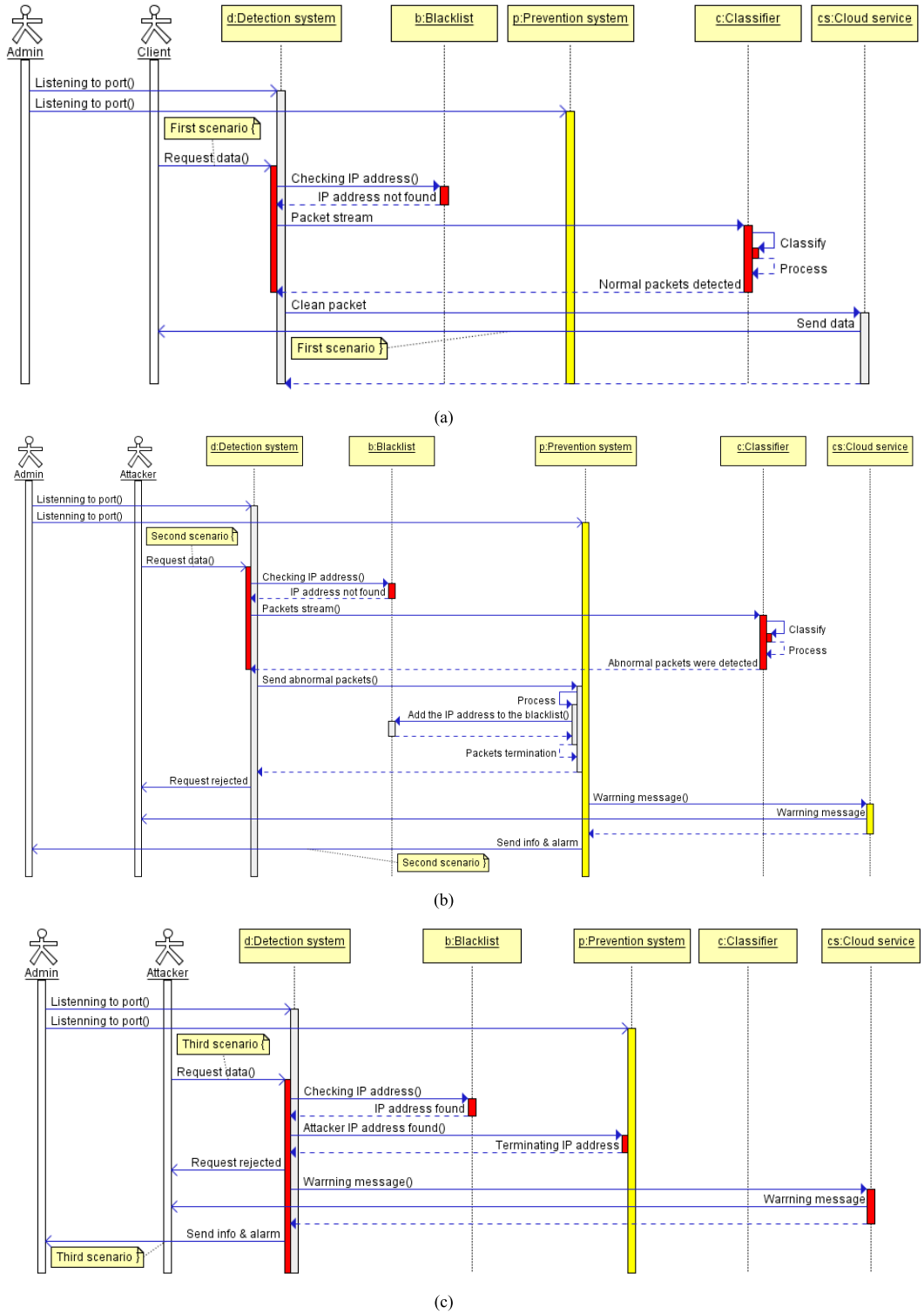


FIGURE 7. CS_DDoS possible scenarios. (a) First scenario (normal packets). (b) Second scenario (store abnormal packets in the blacklist). (c) Third scenario (abnormal packets already in the blacklist).

dimensional and non-linear data. In this work, the LS-SVM is employed to detect illegal activities in a network. The parameters of the LS-SVM are set during the training session to obtain a high proportion of detected results [38].

2) Naïve Bayes

Naïve Bayes is a frequently used classifier and has a straightforward approach based on the application of Bayes' theorem [39]. It is a simple approach which relies on proba-

TABLE 1. Classification performance measurements (n=1000 and K=100).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	99.5%	95.3%	96%	0.91
2	Naïve Bayes	80%	92.3%	93%	0.82
3	K-nearest	75%	93.5%	95%	0.74
4	Multilayer perceptron	88.3%	95.3%	97%	0.78

TABLE 2. Classification performance measurements (n=2000 and K=200).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94.6%	94%	96%	0.89
2	Naïve Bayes	82%	93%	94%	0.75
3	K-nearest	80%	95%	93%	0.87
4	Multilayer perceptron	92%	97%	97%	0.65

TABLE 3. Classification performance measurements (n=5000 and K=300).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	96%	98%	97%	0.90
2	Naïve Bayes	96%	94%	92%	0.82
3	K-nearest	82%	96%	94%	0.68
4	Multilayer perceptron	95%	99%	97%	0.75

TABLE 4. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	99%	98%	0.85
2	Naïve Bayes	95%	95%	96%	0.67
3	K-nearest	85%	98%	97%	0.62
4	Multilayer perceptron	97%	99%	97%	0.58

bilistic knowledge to accurately predict test instances. This algorithm assumes that predictive attributes are conditionally independent and that there are no hidden attributes which can affect the prediction process [39]. The naïve Bayes classifier uses small training sets to provide relatively good performance, which generally overcomes any overtraining issues.

3) K-NEAREST

K-nearest is one of the most straightforward learning algorithms. In this algorithm, the similarity function relies on distance measurements to compute the similarity between training members [40]. The value of k is adjusted during the training session to assign each instance during training to the correct class. The k-nearest classifier is very sensitive to data size and dimensionality, and this affects the feature space and homogeneous areas, which represent the distribution of various classes [41].

4) MULTILAYER PERCEPTRON

The multilayer perceptron is a particular type of neural network-based classifier [42], [43]. This classifier employs a multilayer feed-forward neural network with one or more layers of nodes between the inputs and output layers. These nodes at different layers are interconnected through weighted networks. Using different training algorithms, the parameters (weights) of the networks are optimized. In this classifier, the data are transferred from input to output. Each feature is used as an input in the multilayer perceptron, and the outputs are the class categories. The multilayer perceptron may be linear, when it is used with a single layer of nodes. It can also be a nonlinear perceptron, when it is applied using multiple layers of nodes with several hidden layers [40].

B. PERFORMANCE EVALUATION AND VALIDATION

In this section, the performance of the CS_DDoS system is evaluated and validated using classification performance measurements and K-fold cross-validation.

1) PERFORMANCE EVALUATION

In this section, the performance of the CS_DDoS method is evaluated using the four classifiers of the LS-SVM, naïve Bayes, k-nearest, and multilayer perceptron. Various training data sizes (window sizes) and thresholds are used in the experiments. Algorithm 1 is applied to the training data for all the classifiers.

The CS_DDoS system was evaluated in terms of accuracy, sensitivity (detection rate) and specificity (false alarm

TABLE 5. Classification performance average.

	Classifiers	Average			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	97%	97%	97%	0.8875
2	Naïve Bayes	88%	94%	94%	0.765
3	K-nearest	81%	96%	95%	0.7275
4	Multilayer perceptron	93%	98%	97%	0.69

TABLE 6. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	93%	94%	0.91
2	Naïve Bayes	82%	91.3%	91%	0.82
3	K-nearest	80%	91.5%	92%	0.74
4	Multilayer perceptron	83.3%	92.3%	95%	0.78

TABLE 7. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	93%	91%	94%	0.91
2	Naïve Bayes	85%	92%	95%	0.81
3	K-nearest	79%	96%	92%	0.82
4	Multilayer perceptron	87%	95%	96%	0.71

TABLE 8. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	97%	95%	0.92
2	Naïve Bayes	95%	92%	94%	0.85
3	K-nearest	88%	90%	92%	0.69
4	Multilayer perceptron	89%	97%	93%	0.81

TABLE 9. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	92%	97%	94%	0.87
2	Naïve Bayes	91%	93%	95%	0.65
3	K-nearest	87%	91%	96%	0.69
4	Multilayer perceptron	94%	97%	94%	0.60

TABLE 10. Classification performance average.

	Classifiers	Average			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	95%	94%	0.9025
2	Naïve Bayes	88%	92%	94%	0.7825
3	K-nearest	84%	92%	93%	0.735
4	Multilayer perceptron	88%	95%	95%	0.725

rate), as well as the descriptive statistic Kappa coefficient. Kappa coefficients are procedures used to connect between categorical variables, and are frequently used as consistency or legitimacy coefficients [44].

The accuracy represents the rate of correctly identified results over the entire data used by the CS_DDoS. or true negatives (TN), while incorrectly identified results are false positives (FP) and false negatives (FN). The accuracy of the CS_DDoS system is measured by Equation (1).

- True positives (TP): correctly identified abnormal packets in this research.
- False positives (FP): incorrectly identified abnormal packets.

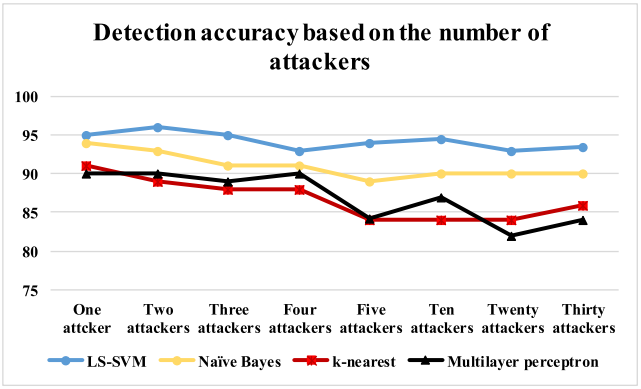


FIGURE 8. Detection accuracy for multiple attacks.

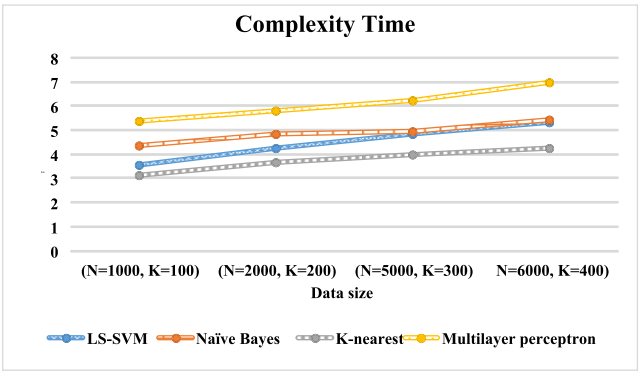


FIGURE 9. Complexity times.

- True negatives (TN): correctly identified normal packets.
- False negatives (FN): incorrectly identified normal packets.

$$CS\_DDoS_{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \quad (1)$$

The sensitivity represents the rate of correctly identified abnormal packets over the entire range of positive results obtained by the CS_DDoS. The sensitivity of the CS_DDoS system is measured by Equation (2).

$$CS\_DDoS_{Sensitivity} = \frac{TP}{TP + FN} \times 100\% \quad (2)$$

The specificity represents the rate of incorrectly identified abnormal packets over the entire range of negative results produced by the CS_DDoS.

The specificity of the CS_DDoS system is measured by Equation (3).

$$CS\_DDoS_{Specificity} = \frac{FP}{FP + TN} \times 100\% \quad (3)$$

The proposed CS_DDoS system is evaluated under both single source and multiple source attack environments, as described below.

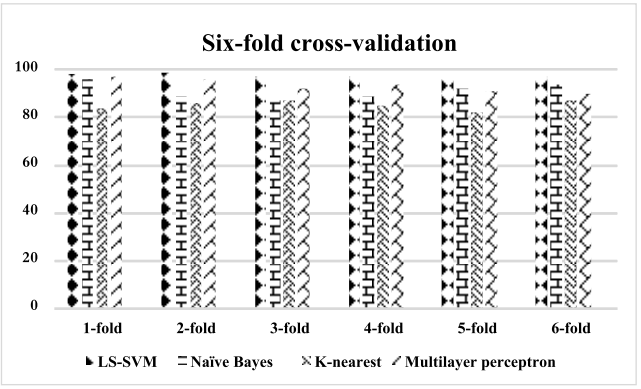


FIGURE 10. Six-fold cross-validation diagram.

a: EVALUATION UNDER SINGLE SOURCE ATTACK

Four test data sizes (n) of 1000, 2000, 5000 and 6000 packets were randomly selected, and four thresholds (K) of 100, 200, 300 and 400 requests. Algorithm 1 was applied to the data according to the window size, n, and was tested according to the threshold K. We have two features fed to each classifier; these two features are the source IP address and the destination IP address. Each classifier was used to classify the data using the four windows and four thresholds. The results are shown in Tables 1-5:

Tables 1 to 4 show the classification performances of the proposed CS_DDoS system with different data sizes and thresholds. The performance measurements are accuracy (correctly detected data over the entire dataset), sensitivity (correctly detected attacks, detection rate), specificity (incorrectly detected attacks, false alarm rate), and Kappa coefficient (stability rate).

According to Tables 1 to 4, the results of each classifier were not significantly affected by the window sizes and thresholds, since there are only small differences between the tables. Tables 1 to 4 are summarized in Table 5.

From Table 5, it can be seen that the LS-SVM classifier has the highest average percentage accuracy (97%) and the highest Kappa coefficient (0.89). Conversely, the k-nearest classifier achieved the lowest accuracy percentage of about 81%, and the multilayer perceptron classifier had the lowest Kappa coefficient 0.69. Overall, the proposed CS_DDoS system is more effective and stable in resisting a single-source attack when adopting the LS-SVM classifier regardless of the window size and threshold.

b: EVALUATION UNDER MULTIPLE-SOURCE ATTACKS

To evaluate the performance under attacks from multiple sources, the same four window sizes were used (1000, 2000, 5000 and 6000) and the same four thresholds (100, 200, 300 and 400). Algorithm 1 was also used. The results are shown in Tables 6–10: Tables 6–9 show the results of the classification accuracy of the proposed CS_DDoS system when under multiple DDoS attacks. Tables 6–9 also show that the results of each classifier were not significantly affected

TABLE 11. Sequence diagram generation codes implemented using quick sequence diagram editor 4.2.

First scenario	Second scenario	Third scenario
Admin:Actor Client:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +1 Client First scenario { +1 (1) Client:d.Request data() d:IP address not found=b.Checking IP address() d:Normal packets detected=c.Packet stream c:Process=c.Classify c:stop d:cs.Clean packet (2)cs:Client.Send data +2 d First scenario } +2	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +3 Attacker Second scenario { +3 (3)Attacker:d.Request data() d:IP address not found=b.Checking IP address() d:Abnormal packets detected=c.Packets stream() c:Process=c.Classify c:stop d:p.Send abnormal packets() p:Packet termination=p.Process p:b.Add IP address to blacklist() d:Attacker.Request rejected p:cs.Warrning message() cs:Attacker.Warrning message (4)p:Admin.Send info & alarm +4 b Second scenario } +4	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +5 Attacker Third scenario { +5 (5)Attacker:d.Request data() d:IP address found=b.Checking IP address() d:Terminating IP address=p.Attacker IP address found() d:Attacker.Request rejected d:cs.Warrning message() cs:Attacker.Warning message (6)d:Admin.Send info & alarm +6 Admin Third scenario } +6

by the window size and the threshold. LS-SVM was again the best performing classifier with percentage accuracy of around 94%, and a Kappa coefficient of about 0.9. Tables 6-9 are summarized in Table 10.

Overall, the proposed CS_DDoS system is also effective and stable in resisting both multiple-source and single-source attacks when using the LS-SVM classifier, regardless of the window size and threshold. Therefore, the proposed CS_DDoS system can be implemented in a large-scale cloud project, such as a health cloud, as well as in smaller projects such as a private cloud for a medium-sized company. CS_DDoS can prevent DDoS attacks with a 94% accuracy and is highly stable (Kappa coefficient 0.9). CS_DDoS outperforms previous approaches, since either the percentage accuracy of previous approaches is lower than those achieved by CS_DDoS, for example 91% in [45], or are without Kappa coefficient stability measurements for example in [46].

In addition, the false alarm rate (specificity) of the benchmark algorithms are 69.57% on average [47]. Thus, we can claim that our proposed CS_DDoS system is more effective.

To shed more light on the performance evaluation of the proposed CS_DDoS system, the simulation was repeated with various numbers of attackers (source IP) under similar conditions and the performance measurements were calculated.

Figure 8 shows the performance of CS_DDoS with an increasing number of attackers. There are slight fluctuations in the performance measurements of all four classifiers, although LS-SVM was still the best performer of the four.

In addition, the process complexity times of the four classification algorithms is shown in Figure 9. While LS-SVM is only the second least time-consuming, the fastest classifier, k-nearest, has lower performance measurements and a smaller Kappa coefficient compared to LS-SVM. It can therefore be considered that the LS-SVM is the most efficient and effective classifier for use in the CS_DDoS system to resist DDoS TCP flood attacks.

2) K-FOLD CROSS-VALIDATION

K-fold cross-validation is a validation model for measuring how the outcomes of a numerical examination will simplify to an independent dataset. Generally, it is utilized to validate the estimation of performance accuracy in practice for a predictive model [48-51].

K-fold cross-validation was used to carry out a performance comparison of the four predictive modeling algorithms used in CS_DDoS: LS-SVM, naïve Bayes, k-nearest, and multilayer perceptron. These four algorithms were compared in terms of their prediction results.

The dataset was divided into six equal-sized chunks, k=6. As a validation for model testing, one of the six chunks was retained, and the remainder (five chunks) were used as training data. Then, the process of the six-cross model was repeated six times, so that each of the six chunks were used as validation data for each model. The results are shown in

Figure 10. We can see that the values of all folds are almost the same, which means that each fold has approximately the same rate for each of the four classification algorithms. Thus, we can claim that the classification results are stable and accurate, since each algorithm gives almost the same results for each fold.

VI. CONCLUSION

The use of cloud computing in many sectors is becoming widespread, as this helps to improve the system in many respects. However, this cloud project is vulnerable to certain types of attacks, such as DDoS TCP flood attacks. Therefore, we propose a new approach called CS_DDoS for the detection and prevention of DDoS TCP flood attacks. The system is based on classification to ensure the security and availability of stored data, especially important for eHealth records for emergency cases. In this approach, the incoming packets are classified to determine the behavior of the source within a time frame, in order to discover whether the sources are associated with a genuine client or an attacker. The results show that using LS-SVM the CS_DDoS system can identify the attacks accurately. The system has an accuracy of about 97 percent with a Kappa coefficient of about 0.89 when under single attack; it is 94 percent accurate with a Kappa coefficient of about 0.9 when under multiple attacks. The performance is validated using K-fold validation and is shown to be stable and accurate. Thus, the proposed approach can efficiently improve the security of records, reduce bandwidth consumption and mitigate the exhaustion of resources. In the future, we aim to extend CS_DDoS to overcome the problem of DDoS using spoofed IP addresses as well as to improve the proposed work to identify the attackers even when they satisfy the threshold value.

APPENDIX

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest regarding the publication of this paper.

REFERENCES

[1] A. Girma, K. Abayomi, and M. Garuba, "The design, data flow architecture, and methodologies for a newly researched comprehensive hybrid model for the detection of DDoS attacks on cloud computing environment," in *Proc. Inf. Technol. Generat. 13th Int. Conf. Inf. Technol.*, 2016, pp. 377–387.

[2] B. Cashell, W. D. Jackson, M. Jickling, and B. Webel, *The Economic Impact of Cyber-Attacks*, document CRS RL32331, Congressional Research Service Documents, Washington, DC, USA, 2004.

[3] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[4] P. A. Laplante, J. Zhang, and J. Voas, "What's in a name? Distinguishing between SaaS and SOA," *IT Prof.*, vol. 10, no. 3, pp. 46–50, May 2008.

[5] C. Balding. (2012). *What Everyone Ought to Know About Cloud Security*. [Online]. Available: http://www.slideshare.net/craigbalding/what-everyone-ought-to-know-about-cloud-security

[6] I. M. Khalil, A. Khreishah, and M. Azeem, "Cloud computing security: A survey," *Computers*, vol. 3, pp. 1–35, 2014.

[7] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, "DDoS attack protection in the era of cloud computing and software-defined networking," *Comput. Netw.*, vol. 81, pp. 308–319, Apr. 2015.

[8] M. Xia, W. Lu, J. Yang, Y. Ma, W. Yao, and Z. Zheng, "A hybrid method based on extreme learning machine and k-nearest neighbor for cloud classification of ground-based visible cloud image," *Neurocomputing*, vol. 160, pp. 238–249, Jul. 2015.

[9] A. Taravat, F. D. Frate, C. Cornaro, and S. Vergari, "Neural networks and support vector machine algorithms for automatic cloud classification of whole-sky ground-based images," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 3, pp. 666–670, Mar. 2015.

[10] A. Sahi, D. Lai, and Y. Li, "Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan," *Comput. Biol. Med.*, vol. 78, pp. 1–8, Nov. 2016.

[11] A. S. Khader and D. Lai, "Preventing man-in-the-middle attack in Diffie–Hellman key exchange protocol," in *Proc. 22nd Int. Conf. Telecommun. (ICT)*, 2015, pp. 204–208.

[12] A. Sahi, D. Lai, and Y. Li, "Parallel encryption mode for probabilistic scheme to secure data in the cloud," in *Proc. 10th Int. Conf. Inf. Technol. Appl. (ICITA)*, Sydney, NSW, Australia, 2015.

[13] A. A. Hameed, B. Karlik, and M. S. Salman, "Back-propagation algorithm with variable adaptive momentum," *Knowl.-Based Syst.*, vol. 114, pp. 79–87, Dec. 2016.

[14] U. R. Acharya *et al.*, "Automated characterization and classification of coronary artery disease and myocardial infarction by decomposition of ECG signals: A comparative study," *Inf. Sci.*, vol. 377, pp. 17–29, Jan. 2017.

[15] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53, 2004.

[16] W. Wei, F. Chen, Y. Xia, and G. Jin, "A rank correlation based detection against distributed reflection DoS attacks," *IEEE Commun. Lett.*, vol. 17, no. 1, pp. 173–175, Jan. 2013.

[17] P. E. Ayres, H. Sun, H. J. Chao, and W. C. Lau, "ALPi: A DDoS defense system for high-speed networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 10, pp. 1864–1876, Oct. 2006.

[18] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An architecture for mitigating DDoS attacks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 1, pp. 176–188, Jan. 2004.

[19] X. Wang and M. K. Reiter, "Using Web-referral architectures to mitigate denial-of-service threats," *IEEE Trans. Depend. Sec. Comput.*, vol. 7, no. 2, pp. 203–216, Apr. 2010.

[20] Y. Xuan, I. Shin, M. T. Thai, and T. Znati, "Detecting application denial-of-service attacks: A group-testing-based approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 8, pp. 1203–1216, Aug. 2010.

[21] K. Salah, K. Elbadawi, and R. Boutaba, "Performance modeling and analysis of network firewalls," *IEEE Trans. Netw. Service Manage.*, vol. 9, no. 1, pp. 12–21, Mar. 2012.

[22] W. Dou, Q. Chen, and J. Chen, "A confidence-based filtering method for DDoS attack defense in cloud environment," *Future Generat. Comput. Syst.*, vol. 29, no. 7, pp. 1838–1850, 2013.

[23] J. Yu, H. Lee, M.-S. Kim, and D. Park, "Traffic flooding attack detection with SNMP MIB using SVM," *Comput. Commun.*, vol. 31, pp. 4212–4219, Nov. 2008.

[24] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim, "DDoS attack detection method using cluster analysis," *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1659–1665, 2008.

[25] A. Chonka, Y. Xiang, W. Zhou, and A. Bonti, "Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks," *J. Netw. Comput. Appl.*, vol. 34, no. 4, pp. 1097–1107, 2011.

[26] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of DDoS attacks for large-scale Internet," *Comput. Netw.*, vol. 51, no. 18, pp. 5036–5056, 2007.

[27] S. Khanna, S. S. Venkatesh, O. Fatemieh, F. Khan, and C. A. Gunter, "Adaptive selective verification: An efficient adaptive countermeasure to thwart DoS attacks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 715–728, Jun. 2012.

[28] R. Guo, H. Yin, D. Wang, and B. Zhang, "Research on the active DDoS filtering algorithm based on IP flow," *Int. J. Commun., Netw. Syst. Sci.*, vol. 7, pp. 600–607, Sep. 2009.

[29] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *Proc. IEEE 35th Conf. Local Comput. Netw. (LCN)*, Oct. 2010, pp. 408–415.

[30] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection," *Pattern Recognit. Lett.*, vol. 51, pp. 1–7, Jan. 2015.

[31] G. Somani, M. S. Gaur, D. Sanghi, and M. Conti, "DDoS attacks in cloud computing: Collateral damage to non-targets," *Comput. Netw.*, vol. 109, pp. 157–171, Nov. 2016.
[32] E. Al-Shaer and S. F. Gillani, *Agile Virtual Infrastructure For Cyber Deception Against Stealthy DDoS Attacks in Cyber Deception: Building the Scientific Foundation*, S. Jajodia, V. S. Subrahmanian, V. Swarup, and C. Wang, Eds., Cham, Switzerland: Springer, 2016, pp. 233–257.
[33] S. Kumar and O. Gomez, "Denial of service due to direct and indirect ARP storm attacks in LAN environment," *J. Inf. Secur.*, vol. 1, pp. 88–94, Jan. 2010.
[34] (Jun. 7, 2016). *Wireshark Analyzer 2.0.0*, [Online]. Available: https://www.wireshark.org/
[35] M. Albanese, E. Battista, and S. Jajodia, *Deceiving Attackers by Creating a Virtual Attack Surface in Cyber Deception: Building the Scientific Foundation*, S. Jajodia, V. S. Subrahmanian, V. Swarup, and C. Wang, Eds., Cham, Switzerland: Springer, 2016, pp. 167–199.
[36] P. Wang, H.-T. Lin, and T.-S. Wang, "An improved ant colony system algorithm for solving the IP traceback problem," *Inf. Sci.*, vol. 326, pp. 172–187, Jan. 2016.
[37] (Jul. 20, 2016). *Quick Sequence Diagram Editor 4.2*, [Online]. Available: https://github.com/sdedit/sdedit
[38] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.
[39] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell.*, 1995, pp. 338–345.
[40] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Hoboken, NJ, USA: Wiley, 1973.
[41] A. Depeursinge *et al.*, "Comparative performance analysis of state-of-the-art classification algorithms applied to lung tissue categorization," *J. Digit. Imag.*, vol. 23, no. 1, pp. 18–30, 2010.
[42] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, vol. 4, no. 2, pp. 4–22, Apr. 1987.
[43] R. K. Madyastha and B. Aazhang, "An algorithm for training multilayer perceptrons for data classification and function interpolation," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 41, no. 12, pp. 866–875, Dec. 1994.
[44] H. C. Kraemer, V. S. Periyakoil, and A. Noda, "Kappa coefficients in medical research," *Statist. Med.*, vol. 21, no. 14, pp. 2109–2129, 2002.
[45] M. Moradi and M. Zulkernine, "A neural network based system for intrusion detection and classification of attacks," in *Proc. IEEE Int. Conf. Adv. Intell. Syst.-Theory Appl.*, Nov. 2004, pp. 1–6.
[46] R. Jalili, F. Imani-Mehr, M. Amini, and H. R. Shahriari, "Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, 2005, pp. 192–203.
[47] A. Kuzmanovic and E. W. Knightly, "Low-rate TCP-targeted denial of service attacks: The shrew vs. the mice and elephants," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2003, pp. 75–86.
[48] G. McLachlan, K.-A. Do, and C. Ambroise, *Analyzing Microarray Gene Expression Data*, vol. 422. Hoboken, NJ, USA: Wiley, 2005.
[49] D. M. Allen, "The relationship between variable selection and data agumentation and a method for prediction," *Technometrics*, vol. 16, no. 1, pp. 125–127, 1974.
[50] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. Roy. Statist. Soc. B (Methodol.)*, vol. 36, no. 2, pp. 111–147, 1974.
[51] S. Geisser, "The predictive sample reuse method with applications," *J. Amer. Statist. Assoc.*, vol. 70, pp. 320–328, Jun. 1975.

AQEEL SAHI received the bachelor's degree in computer science from Thi-Qar University, Iraq, in 2007, and the master's degree in information technology from University Utara Malaysia, Malaysia, in 2010. He is currently pursuing the Ph.D. degree with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia. His current research interests are in cryptography and parallel processing with a focus on block cipher modes of operation and key exchange protocols.

DAVID LAI received the B.Sc., PGDipEd, and M.Phil. degrees from The Chinese University of Hong Kong, the GDipCompSc degree from the Vaal University of Technology, the MIT degree from the Queensland University of Technology, and the Ph.D. degree from the University of Southern Queensland.

He is currently a Senior Lecturer with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia.

YAN LI received the B.Eng. and M.Eng. degrees from the Huazhong University of Science and Technology, and the Ph.D. degree from Flinders University.

She is currently an Associate Professor with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia.

She is also a Approved Research Supervisor in the area of signal processing (090609), computer communications, networks (100503), fields of research, biomedical engineering, artificial intelligence, image processing, signal processing, and computer communications networks.

Her research interests are machine learning algorithms, big data analytics, signal/image processing, EEG research, graph theory, and networking technologies.

MOHAMMED DIYKH received the B.Sc. degree in computer science from Thi-Qar University, Iraq, in 2003, and the M.Sc. degree in information technology from Voronezh State University, Russia, in 2010. He is currently pursuing the Ph.D. degree with the Faculty of Health, Engineering and Sciences, University of Southern Queensland, Australia. His current research interests include biomedical signal analysis, data mining, and graph theory.