A new framework to alleviate DDoS vulnerabilities in cloud computing

A.Saravanan[1], S.SathyaBama[2], Seifedine Kadry[3], Lakshmana Kumar Ramasamy[4]
[1]Department of Computer Science and Applications, Sree Saraswathi Thyagaraja College, Pollachi, India
[2]Independent Researcher, Coimbatore, India
[3]Department of Mathematics and Computer Science, Faculty of Science, Beirut Arab University, Lebanon
[4]Department of MCA, Hindusthan College of Engineering and Technology, India

Article Info

Article history:

Received Jan 14, 2019
Revised Mar 16, 2019
Accepted Apr 27, 2019

Keywords:

Alleviate framework
Cloud security
Distributed denial of service
Queuing model
System availability

ABSTRACT

In the communication age, the Internet has growing very fast and most industries rely on it. An essential part of Internet, Web applications like online booking, e-banking, online shopping, and e-learning plays a vital role in everyday life. Enhancements have been made in this domain, in which the web servers depend on cloud location for resources. Many organizations around the world change their operations and data storage from local to cloud platforms for many reasons especially the availability factor. Even though cloud computing is considered a renowned technology, it has many challenges, the most important one is security. One of the major issue in the cloud security is Distributed Denial of Service attack (DDoS), which results in serious loss if the attack is successful and left unnoticed. This paper focuses on preventing and detecting DDoS attacks in distributed and cloud environment. A new framework has been suggested to alleviate the DDoS attack and to provide availability of cloud resources to its users. The framework introduces three screening tests VISUALCOM, IMGCOM, and AD-IMGCOM to prevent the attack and two queues with certain constraints to detect the attack. The result of our framework shows an improvement and better outcomes and provides a recovered from attack detection with high availability rate. Also, the performance of the queuing model has been analysed.

Corresponding Author:

Seifedine Kadry,
Department of Mathematics and Computer Science,
Faculty of science,
Beirut Arab University, Lebanon.
Email: s.kadry@bau.edu.lb

1. INTRODUCTION

Cloud computing is an emerging field that provides virtual solutions to the business and other users. It offers several services to customers with unique features like improved scalability, availability and manageability of resources on demand. The user’s data and applications are stored on cloud storage and the users can access them at anywhere and at any time. Even though the cloud environment has various advantages, it has various risks and challenges in the case of security, since it stores the user’s personal, confidential and critical data. Thus, trustworthiness, service availability and sensitive data protection are the important concerns for cloud users. International Data Corporation (IDC) conducted a survey in August 2008 with the cloud users. According to the survey, security is the major barrier in using cloud environment [1]. The cloud service providers should guarantee connectivity, availability, and security to the cloud users. If this guarantee is compromised, then the user or the organization will suffer ridiculously [2]. Denial of Service (DoS) attacks and Distributed Denial of Service (DDoS) attacks are two main network centred vulnerabilities targeting the web servers and the cloud servers to make resource inaccessible to authentic users. The presence of a DDoS attack was identified in June 1998 and this is considered as the first occurrence in web history [3]. As the data in the cloud is available to the legitimate users at any time and since it can be accessed from anywhere, this attack is increasing every year. Since the data are distributed in cloud and web servers, several network attack tools are developed and readily available to instigate the attack. Ref. [4] is considered to be the first tool which has a set of programs written in C. Generally, this tool was used widely by hackers to launch DDoS attacks. Several DDoS attack tools are currently available to simulate enormous packet requests concurrently to victim server, which provides unavailable service to the legitimate users. However, due to the improvement in technology, botnets are introduced to commence the DDoS attacks. The malware is planted in the group of systems which becomes zombie or botnet that instigate simultaneous request to the victim to create the attack traffic. If this attack continues for a longer period, it even excludes web spiders and web crawlers from visiting the website which leads to the reduction in page ranking for the particular site. So, the user may not show interest in visiting the site, since the pages are not shown by the search engines due to low ranking [5].

The objective of this paper is to detect and analyse Distributed Denial of Service (DDoS) attacks in a cloud computing environment. The proposed framework can be used instead of image reCAPTCHA and other methods along with no CAPTCHA. The proposed prevention strategies identify bots from humans through screening tests which can be performed after no CAPTCHA reCAPTCHA and the detection strategies mitigates the DDoS attacks such as SYN flood, ACK flood, Layer 7 DDoS attack, Smurf attacks, and others. The performance of the system is compared with other methods and proves to be better with a detection rate of 98%.

2. DISTRIBUTED DENIAL OF SERVICE ATTACK

Mostly, in this type of attack, the target system is flooded with incoming messages and forces the system to move to a busy state, thereby denying service to legitimate users. A typical DDoS attack scenario is shown in Figure 1.

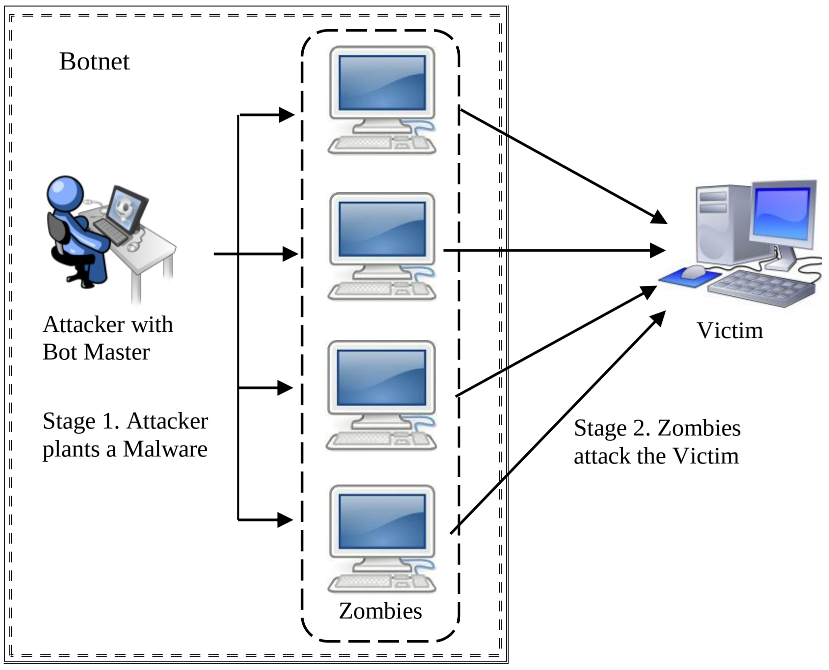


Figure 1. A typical scenario for distributed denial of service attack

For a successful DDoS attack, the attacker uses a two stage process [6]. In the first stage, the attacker exploits a vulnerability and plants a Trojan horse or any malware on a target machine. The malware may not be noticeable since it does not cause any harm to the target system. Now, the target machine becomes the DDoS master or botmaster. Either the botmaster or the attacker identifies other vulnerable systems and infects them with the malware. Each of these systems now becomes zombie or bot. This group of bots is called a botnet. Nevertheless, these systems carry out their normal work since they are unaware of the resident zombie. In due course, the attacker identifies a victim and with the help of botmaster, sends a signal to all the zombies to launch the attack on the victim. Now, the victim encounters a number of attacks from all zombies at the same point of time. Moreover, the zombies may not use the same attack. Each zombie may use different flooding attacks. As in [5], DDoS attacks are categorized into two broad categories:

- Network Centric DDoS attack (Layer-3 attack)
- Application Centric DDoS attack (Layer-7 attack)

Generally, network and transport centric DDoS attacks are carried out to exhaust server's resources by arraying an enormous number of packets of TCP, UDP, ICMP protocols. These are named as flood attacks. Layer 7 attack exploits the vulnerabilities of application level protocols and depletes victim server's resources using HTTP and other applications.

3. RELATED WORK

The defense against DDoS attack can be provided at various stages. Somani et al. categorized the defense against DDoS as attack prevention, attack detection and attack recovery [7]. Figure 2 depicts the defense mechanism against the DDoS attack. A detailed survey about the detection, prevention, and recovery methods with their pros and cons has also been described by them [8].

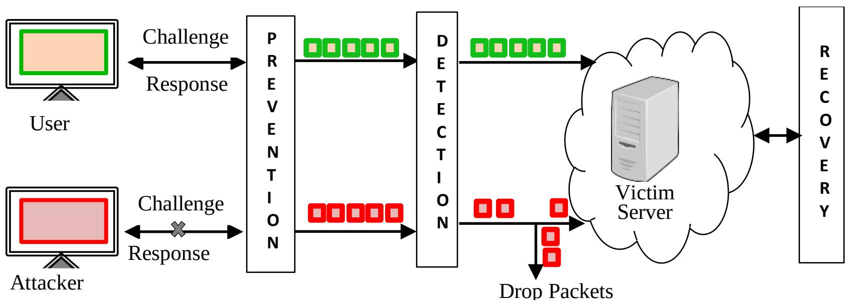


Figure 2. Defense mechanism against the DDoS attack

As a preventive measure, "Graphical Turing Test" can be used to distinguish machines from a human. These are intended to be unreadable by intelligent machines, or even the screen readers cannot understand them. Basically, the use of challenge response system may help in preventing the attack. One of the most common implementations as a prevention strategy is a Turing Test in the form of CAPTCHA images [9]. This protocol is usually considered as one of the most ideal methods in the class of challenge response systems. It tries to perceive whether the user is a bot/attacker machine. This type of protocol may also include graphical test [10], Text Puzzles [11], Crypto Puzzles and Proof-of-Work [12] to prevent the bot driven attack to occur. Several distortion or noise, such as waviness and horizontal stroke were also added to escalate the complexity of breaking the CAPTCHA with a computer program. The reCAPTCHA and image reCAPTCH which is an enhancement of CAPTCHA supplies the websites with images of words that are hard to read for optical character recognition (OCR) software, as a challenge to the clients. Image identification CAPTCHA is also widely used in recent days. Various methods like Naming CAPTCHA and Anomaly Detection CAPTCHA are also in use. The main downsides of the above methods are the graphics generation and storage space overhead. Conversely, text puzzles can also be used to identify the bot system, but the limitation is OCR attacks. Apart from challenge response system, other methods have also been proposed to prevent the DDoS attack with restricted access [13].

However, several methods have the possibility of puzzle accumulation attack and even puzzle generation and space to store the images are additional overhead. Client puzzle mechanism is implemented in [14]. The second stage is an attack detection. With the help of some detection methods, the attack signatures are detected. The attack may be at the initial stage without any injection or it may be at the final stage where it has already affected the system. Several methods have been proposed using anomaly detection [15], web behavior [16], trace back method [17], threshold filtering which includes hop count [18], request count [19] and confidence based filtering [18]. A Queue model to detect the DDoS attack is proposed [20]. The covariance analysis model has been also implemented to detect the attack [21].

To detect modern botnet-like malware based on anomalous patterns in a network, entropy-based network anomaly detection method is explained in [22].

The authors Lonea, et al. proposed a method that combines the report given by the intrusion detection systems deployed in virtual machines with a data fusion approach [1]. But many of these methods do not have support for efficiency and some methods are not scalable. The third stage is a recovery stage in which the methods are to implement in the victim server to serve its user or to recover from the attacks. Several methods including migration [23] and backup resources [24] are suggested by a few researchers. But at this stage, implementation of any method leads to an overhead of reserved resources and costs to the victim server. Recently, a protection policy to dynamically install security applications across the controller and switches has been proposed by Han et al. that identifies the accurate location of the botnet [25]. Support Vector Machine (SVM) algorithm has been used in creating the model for classification of DoS attacks and normal network behaviors [26]. The survey on various distributed denial-of-service attack, prevention, and mitigation techniques has been discussed in Mahjabin et al. [27]. Thus, from the literature survey, it is clear that the main limitations in prevention strategies are graphics, image, puzzle and text generation as well as the storage space overhead. Additionally, some methods pave the way for other attacks such as OCR and puzzle accumulation attacks. In case of attack detection, few methods do not support more efficiency and scalability. In the last case of attack recovery, the reserved resources and the costs to the victim server are more.

4. PROPOSED METHOD

The main goal of the system is to mitigate the DDoS attack by providing the preventive and detective measures. The overall process of the proposed system is given below in Figure 3. Queuing Model is used on the server side. A Finite Capacity Markovian Queuing Model M/M/1/K is used [28]. It is a single server queue with a queue size K. The server has two queues, Processing Queue and Waiting Queue. All the request from the client is verified and it is stored in the processing queue. If the processing queue is full and if certain conditions met, the request will be stored on the waiting queue. The main goal of introducing the waiting queue is to process all the packets later, instead of dropping the suspicious packets. Also based on the number of packets in the queue, the prevention mechanism varies. The three enhanced CAPTCHA mechanisms are introduced namely VISUALCOM, IMGCOM, AD-IMGCOM. If the processing queue is full, then the intricate method IMGCOP or AD-IMGCOM is given as a challenge to the user, else the simple method VISUALCOM is used.

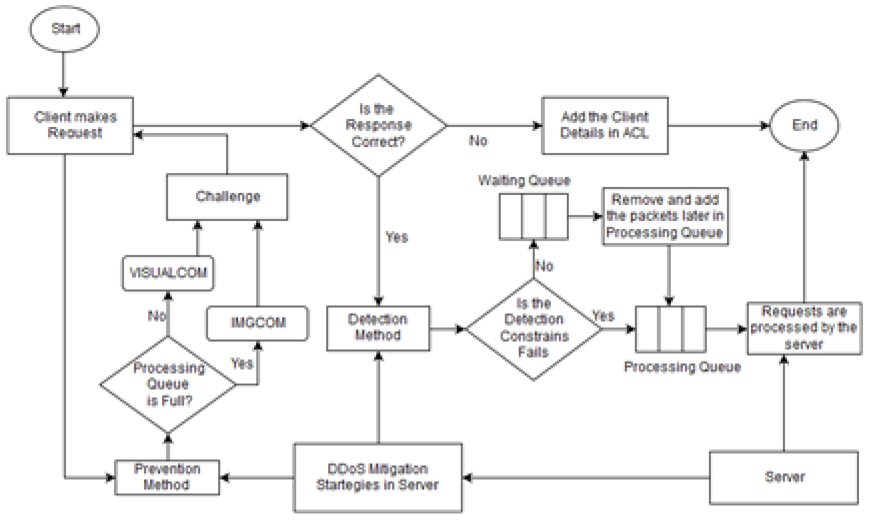

Start
Client makes Request
Is the Response Correct?
No
Add the Client Details in ACL
End
Challenge
Yes
Waiting Queue
Remove and add the packets later in Processing Queue
VISUALCOM
No
IMGCOM
Detection Method
Is the Detection Constrains Fails
Yes
Requests are processed by the server
No
Processing Queue is Full?
Yes
Processing Queue
Prevention Method
DDoS Mitigation Startegies in Server
Server

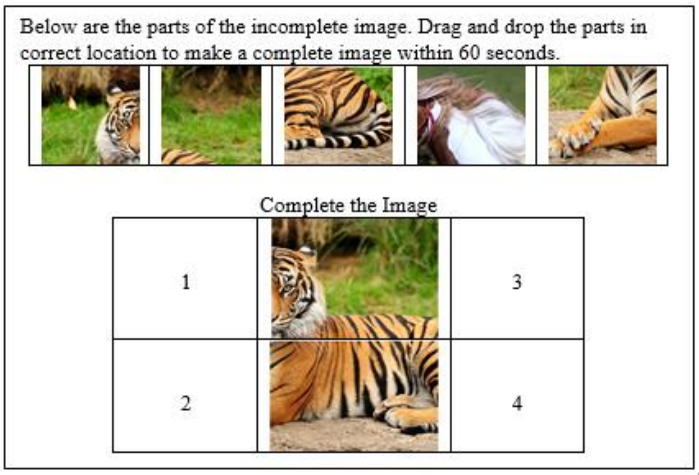
Figure 3. The overall architecture of the mitigation framework for DDoS vulnerabilities

4.1. Queuing model

Two queues are proposed in this framework. All the packets or requests arrived at the server are checked with detection constraints. Initially, the packets waiting to be processed are stored in a processing queue. However, if any of the constraints met or if the processing queue is full, then the requests are stored on the waiting queue. Later, the requests are moved to the processing Queue for further process. Thus, all the requests are processed and even a suspicious request is also processed but with some delay or when the server is idle. The processing queue is finite with Poisson arrival and exponential service and the waiting queue is infinite. There are some assumptions to be followed in implementing the queuing model. The servicing method can accept and store k requests merely. The arrival rate of a requests (the number of requests arrived per unit time) is denoted by λ. The effective arrival of the request (the rate of requests entering in the system) is given by λ_e. These requests are stored in the queue. k is the maximum number of services in the system. The rate of the request not entering the queue is given by λ_b. However, these requests are stored in the waiting queue. Thus $\lambda=\lambda_e+\lambda_b$. The service rate of the requests (the number of requests serviced per unit time) is denoted by μ. An average number of requests in the system is given by L_s. The number of requests in the queue is given by L_q. The average waiting time of the request before completion of its request is given by W_s. The requests are processed in First Come First Served scheduling. In the queue, the requests wait for a time W_q for the service. ρ is the utilization factor which determines the proportion of time that the server is busy servicing requests [29].

4.2. Prevention strategy

Since sites may use CAPTCHAs as part of the preliminary registration procedure, or as a part of every login process, the challenge can completely block access to the machines by distinguishing them from a human. This section explains about three tests namely VISUALCOM, IMGCOM, and AD-IMGCOM. Visual comprehension (VISUALCOM) can be applied by providing the visual scene or picture to the user. Then the questions regarding the visual scene can be given as a challenge to the user. The user gives the answer as a response. The main advantage of this method is that storage space. In normal Graphical Turing Test, each test needs one or more images which increases the space complexity. Thus ‘n’ users need ‘n’ or more than n images to undergo their Turing Test. However, in this proposed method, with a single image, four to five questions can be framed. So a single image can serve ‘m’ users where m is the number of framed questions with a particular image. The next proposed method, Image Completion (IMGCOP) is complex when compared to the VISUALCOM. In this method, a single image is divided into parts. The incomplete image along with the parts are given as a challenge to the user. The user is supposed to drag and drop the parts to make a complete image. If the image matches with the one in the database, then the user is authenticated. As like the existing system, a single image is used, but the complexity is increased through which even the intelligent bot cannot give a correct response. Another method, which is a variation of IMGCOP is Image Completion with Anomaly detection (AD-IMGCOM). It is similar to the IMGCOM, where the parts of the image are given with additional anomaly. Thus the user has to identify the anomaly and also drag and drop the parts correctly to make the original image. This is shown in Figure 4.


Below are the parts of the incomplete image. Drag and drop the parts in correct location to make a complete image within 60 seconds.

Complete the Image

Figure 4. Sample AD-IMGCOM Turing Test to identify the human and a bot with 5 parts and an incomplete image. The fourth part is an anomaly

4.3. Detection strategy

Even after the challenge response verification, there is a possibility for flooding attack. Thus, this section provides the detection methods to identify the attacks. Each packet is analyzed to identify the attack.

Several parameters are used to analyze the packets like IP address, TTL, etc.. Three variables Request Count C_R, Source Count C_S, SYN Flag Count C_F are maintained by the server. The request count for each source is stored in C_R. Similarly, new packets arriving continuously from different sources is stored in C_S. Next, the packets arriving continuously with SYN flag on is maintained in C_F to monitor SYN flood attack. Also, the threshold value for each variable is set to identify the DDoS attack. The following are various conditions that are to be checked to move the packets to the processing queue and waiting queue.

- A Time To Live (TTL) is an eight bit field to specify the maximum lifetime of an IP packet. From source to destination, the packet will pass through several routers and each router decreases the TTL value of an IP packet by one. As in [30] the server maintains IP2HC table. Each arrived packet is verified with the IP2HC table with the calculated HC and in Access Control List (ACL), if there is a match, then the packet undergoes the next security check, else it is spoofed. In such case, the packets can be added to the waiting queue and the ACL can be updated with the particular IP address.
- If several packets arrived from the same source, then the count (request count C_R) will be maintained. If the count exceeds the threshold value T_R, then the packets are stored in the waiting queue else it undergoes the next check.
- Similarly, if several new packets arrived continuously from different sources, again the count called source count C_S, is maintained and updated. Again if it exceeds the threshold T_S then the packets are moved to the waiting queue.
- The next parameter is SYN flag. If several packets arrived continuously with SYN flag on, the count C_F will be maintained. If the count exceeds the threshold T_F then the packet will be moved to the waiting queue.

The packets stored in the waiting queue are processed after a minimum delay. The detection algorithm is shown in Figure 5.

```
For each packet
Extract the IP address, TTL, SYN flag;
Update Request Count $C_R$, Source Count $C_S$, SYN
Flag Count $C_F$;
$CH_1$ = Compute HC from the TTL;
$CH_2$ = Access HC for the IP address from IP2HC;

If ($CH_1 = CH_2$) then
  Else if ($C_R < T_R$) then
     Else if ($C_S < T_S$) then
              Else if ($C_F<T_F$) then
                 Store the packet in the Processing Queue
Else
                 Store the packet in the Waiting Queue
End If
End For
```

Figure 5. Proposed detection algorithm

4.4. Construction of IP2HC table

The mapping between the IP address and the Hop Count is maintained by the server. The Border Gateway Protocol generally maintains the HC to other hosts for which it needs to communicate. Whenever it receives the packet from the particular IP address it verifies with the IP2HC table. If there is no match for the particular IP address, it broadcasts the Route Request RREQ packet to the neighbours with the particular IP address as the Destination Address (DA). On receiving the request, the neighbours verify the DA with its own IP address. If the destination address is not their own, they further forward to the neighbouring routers. The intended host, on receiving the request, sends the Route Reply RREP packet containing the route, hop count and other information to the source in the same route as RREQ but in reverse direction. Also, as in DMIPS [31], a verification step can be carried out by sending a query to the host and by setting the retrieved Hop Count as the TTL value. After the reply has arrived, the source then updates the IP2HC table with the IP address and Hop Count.

Once the packet has arrived, the IP address is matched with the IP2HC table. If there is a match, the HC will be calculated from the TTL value and compared with the Hop Count in the table referred as CH_2. The Hop Count value can be directly calculated from the TTL value of the received packet as the intermediary router decreases the TTL value of the packet before forwarding it to the subsequent router. Since, the attacker cannot modify the values of the number of hops required for a packet to reach its destination, even though there is a possibility of modifying the fields in the IP header. The calculation of Hop Count is given in (1).

$$\text{Hop Count } CH_1 = \text{Initial TTL Value} - \text{Final TTL Value} \quad (1)$$

The Final TTL Value is the one extracted from the received packet. The receiver calculates the Initial TTL Value which is more challenging. Luckily, maximum modern OSs employ only a limited and certain initial TTL values, 30, 32, 60, 64, 128, and 255. And since the maximum number of hops between any two nodes on the internet is more than 30 hops, the receiver can calculate the Initial TTL Value as the smallest value that is larger than the Final TTL Value. For example, if the Final TTL Value is 108, then the Initial TTL Value will be minimum of (128, 255) which is 128. The proposed method detects various DDoS attacks in the cloud server. If several packets arrived from the same source, and if it exceeds the threshold value, the packets are moved to the waiting queue. This sets a limit for each client a configurable number of request to the server which detects layer 7 application layer DDoS attacks. Also if several new packets arrived continuously from different sources and the count exceeds the given threshold, the packets are moved to the waiting queue. This conditions will detect and mitigate the smurf attacks, ACK flood attacks. Generally, these two conditions detect varies lsyer 3, layer 4 and layer 7 DDoS attacks. The SYN flood attacks can be detected and mitigated by monitoring the count of the packets arriving with SYN flag on. Thus the proposed method outperforms in mitigating DDoS vulnerabilities.

5. RESULTS AND ANALYSIS

The evaluation of any detection method is extremely important before deployment in a real-time network. Thus the experimental analysis uses a single server and 15 clients to generate the network traffic and attack traffic. Netwag tool [32] is used to produce the known DDoS attacks such as a TCP SYN attack, smurf attack etc. Additionally, to identify the packets and access all its header information, a packet capturing tool JPCap is employed [33]. Generally, for transmitting and capturing the packets from the network, JPCap will be a perfect choice which is an open source java Library. The legitimate traffic has been used as a training data with 100 DDoS attacks. The overall prevention and detection rate for the proposed method along with the existing methods such as pushback [34], distance based and C_2DF [33] have been analysed and the comparison is shown in Figure 6.

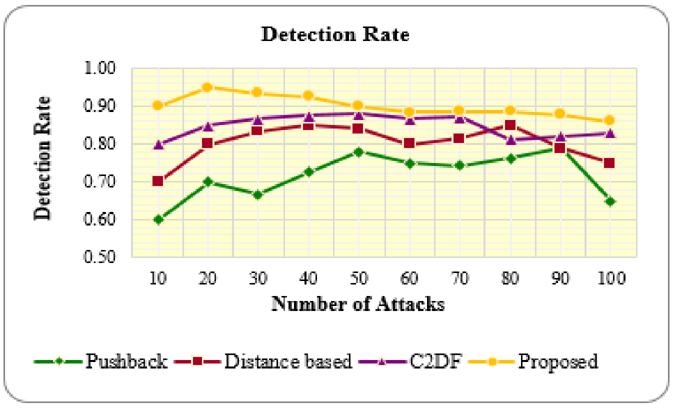


Figure 6. Performance Analysis with DDoS Detection Rate

The existing Pushback method is less effective since it uses high computation power and execution time. The distance based method does not perform effectively in case of recovery phase and resource utilization. The settings of C_2DF method suffers from low accuracy in detecting the DDoS attacks. The graph in Figure 7 shows the comparison of the false negative rate for all the existing methods and proposed a DDoS mitigation approach. Also, the analysis has been made to compare the dropout rate for the proposed and existing algorithms by varying the number of requests.

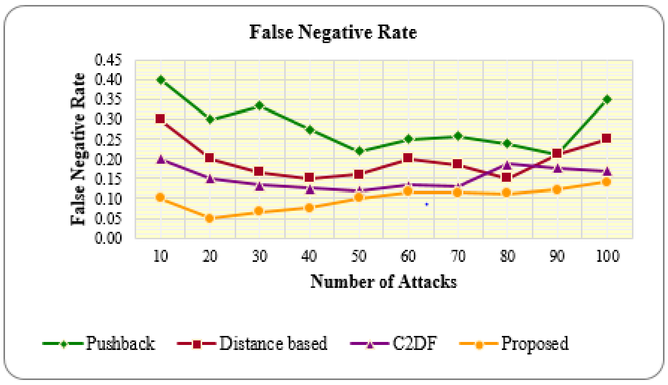


Figure 7. Performance analysis with DDoS false negative rate

The drop rate is the fraction of the number of packets dropped to the total number of packets. Figure 8 represents the dropout rate comparison. From the analysis, it is clear that the dropout rate is minimum for the proposed mitigation framework when compared with the existing methods.

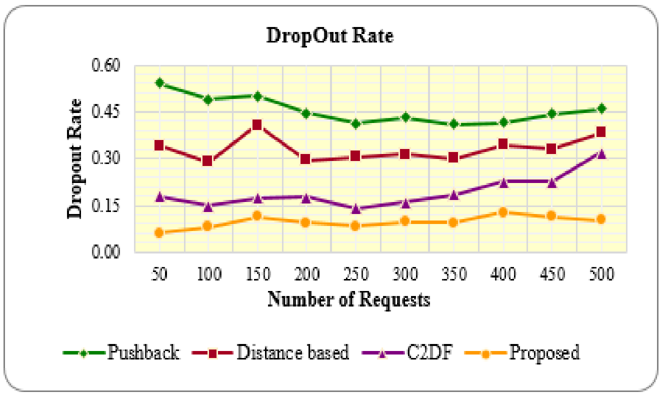


Figure 8. Performance Analysis with DDoS Dropout Rate

Also, the performance evaluation has been made based on the execution time for the proposed model as a parameter. The time to fetch the images and to verify the response is analyzed for the proposed method. Table 1 shows the analysis of challenge response system with the execution time of three methods. Hence, the average execution time for VISUALCOM is 29.2 ms, that of IMGCOP is 42.6 ms and AD-IMGCOP is 48.6 ms. Without the proposed prevention techniques, the execution time of the system is 14.4 ms.

Table 1. Comparison of execution time with and without proposed prevention methods

	Execution Time in milliseconds				
Test No	With VISUALCOM	With IMGCOP	With AD-IMGCOP	Without Prevention techniques	With VISUALCOM
1	25	41	46	12	25
2	31	48	53	14	31
3	27	39	45	18	27
4	29	42	48	15	29
5	32	46	49	16	32
6	34	41	50	14	34
7	27	42	51	15	27
8	28	43	48	13	28
9	30	42	47	12	30
10	29	42	49	15	29
Average	29.2	42.6	48.6	14.4	29.2

The average time delay is of 14.8, 28.2 and 34.2 milliseconds respectively, for the proposed methods which are negligible when the consequence of the DDoS attacks is considered. The graph in Figure 9, clearly depicts the time taken for the proposed method.

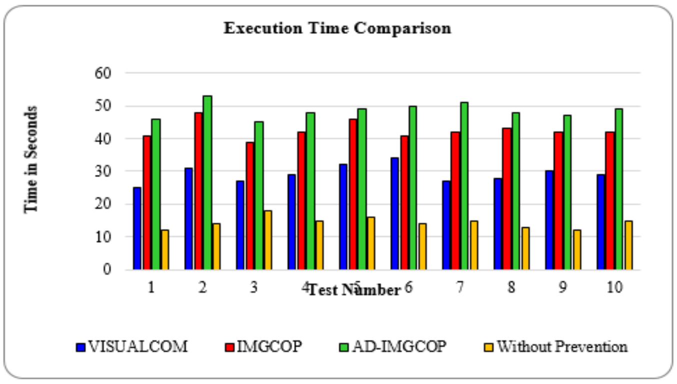


Figure 9. Comparison of execution time for prevention methods

Among the three prevention mechanism AD-IMGCOP takes more time and thus it is alone compared with other strategies such as reCAPTCHA and image reCAPTCHA. From Figure 10, it is clear that the execution time is minimal when compared with the other existing techniques. Also, the proposed methods require less storage space when compared with the others and it is discussed at the end of the section. Thus if no CAPTCHA is not too sure, then the proposed method can be used instead of other existing techniques.

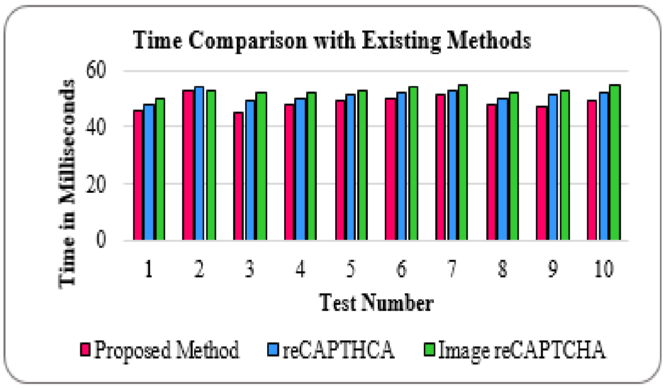


Figure 10. Comparison of execution time for prevention method with existing methods

To find out the performance of the processing queue, the analysis has been made by changing the size of the queue and the average waiting time of the system is calculated. The Arrival rate of packets is $\lambda=9$ packets/min. The average service time for a single packet is 10 seconds. Then the service rate $\mu=1/10$ packets/sec=6 packets/min. thus the Utilization value $\rho=9/6 = 1.5$. The calculation of a number of requests in the queue, waiting time in the queue and the time taken to complete the request are given in Table 2.

Table 2. Waiting time and completion time calculation for various queue size

Queue Size	Effective arrival rate (packets/min)	No of requests waiting in queue	Time to complete the service (in min)	Waiting time in the queue (in min)
5	5.8	3.4	0.76	0.59
10	5.9	8	1.5	1.3
15	5.9	13	2.3	2.1
20	6	18	3.1	3
25	6	23	4	3

Thus when the queue size is small, the performance of the system is better, since the waiting time is small for small queues than the long queues. The performance analysis is shown in Figure 11. The queue size is fixed at 10. The proposed framework has a capability to detect above 95% of attacks. However, when the threshold is decreased, the accuracy is increased even better. This is shown in Table 3. The table shows the accuracy and error rate with the threshold value between 5 and 10.

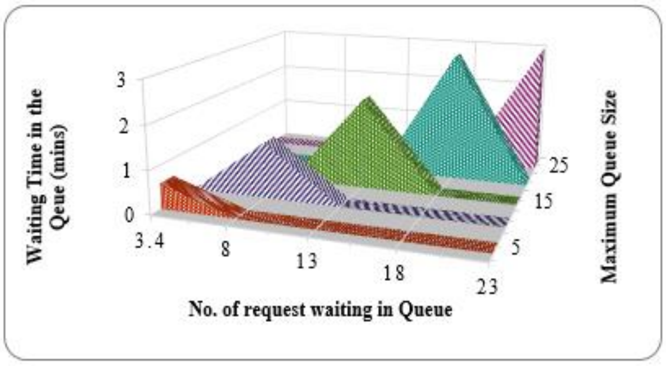


Figure 11. Performance analysis of the queue with varied size

Table 3. Waiting time and completion time calculation for various queue size

Threshold Value	Correctly classified attacks	Incorrectly classified attacks
10	95	5
9	95.4	4.6
8	96.2	3.8
7	97	3
6	97.8	2.2
5	98.6	1.4

The accuracy and error rate for the above implementation is shown graphically using the chart in Figure 12. Also, the proposed method is analyzed with memory space as another parameter. For VISUALCOM, single image with 5 questions for each image is taken into account which can serve 5 users.

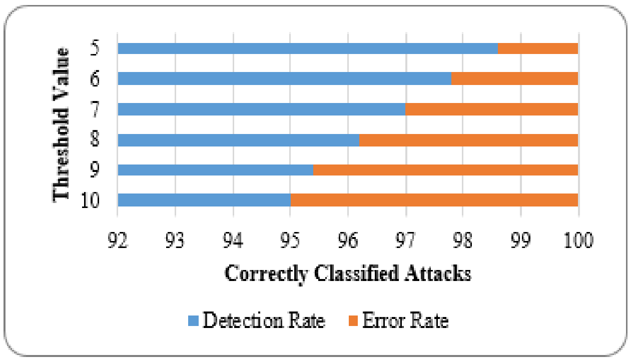


Figure 12. Result analysis with varying threshold values

Similarly, IMGCOP stores a complete image with its part. In our experiment, each image is split into 6 parts and given as a challenge to the client. However, a single image can serve up to 15 users with different combinations. The third method AD-IMGCOP needs the same memory as IMGCOP along with small additional memory to store the anomaly image as well. The existing method used for comparison is that an image reCAPTCHA in which a group of image that is selected based on the given challenge. This experimental result is shown in Table 4. Figure 13 shows the graphical representation of the comparison.

Accordingly, the proposed methods require low memory space when compared to existing image reCAPTCHA.

Table 4. Comparison of space requirement for prevention methods

No of Users	Storage Memory in Kilo Bytes			
	VISUALCOM	IMGCOP	AD-IMGCOP	Image reCAPTCHA
10	44	68	72	240
20	88	136	144	480
30	132	204	216	720
40	176	272	288	960
50	220	340	360	1200

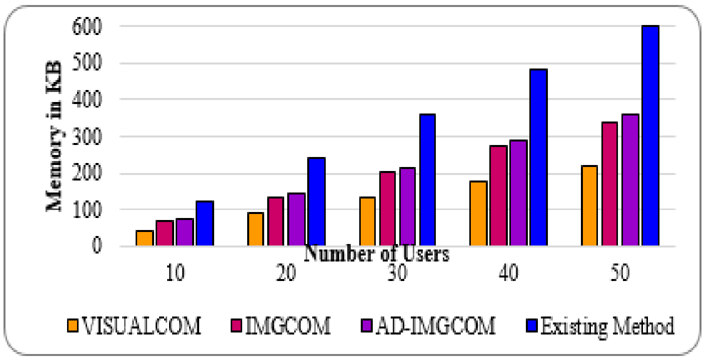


Figure 13. Comparison of memory space in kilobytes for prevention methods

From the evaluation, it is clear that the prevention strategies proposed are stronger for the botnets to execute the DDoS attacks. Also, the memory requirement is minimum. The execution time is increased by the small amount which can be negligible.

6. CONCLUSION

In this paper, the DDoS vulnerabilities are discussed along with the possible defense mechanism. Also, a framework has been proposed which includes prevention and detection mechanism. It uses the enhanced Graphical Turing tests to prevent the attacks from a botnet. Additionally, an algorithm that analyzes the packet header is added to the framework. This system is implemented with a queuing model. The experimental results are evaluated with a fixed queue size. In the near future, a mechanism to modify the queue size dynamically can be introduced by evaluating the waiting time. Also, the future work aims at recovery techniques like migration.

REFERENCES

[1] A. M. Lonea, et al., “Detecting DDoS attacks in cloud computing environment,” *International Journal of Computers Communications & Control*, vol/issue: 8(1), pp. 70-78, 2013.

[2] G. Perry, “Minimizing public cloud disruptions,” TechTarget, 2011. Available: http://searchdatacenter.techtarget.com/tip/Minimizing-public-cloud-disruptions.

[3] S. C. Lin and S. S. Tseng, “Constructing detection knowledge for DDoS intrusion tolerance,” *Expert Systems with applications*, vol/issue: 27(3), pp. 379-390, 2004.

[4] D. Dittrich, “The DoS Project's "trinoo" distributed denial of service attack tool,” 1999. Available: http://staff.washington.edu/dittrich/misc/trinoo.analysis.

[5] N. C. S. Iyengar, et al., “A fuzzy logic based defense mechanism against distributed denial of services attack in cloud environment,” *International Journal of Communication Networks and Information Security (IJCNIS)*, vol/issue: 6(3), 2014.

[6] S. Deshpande and R. Ingle, “Preferences Based Customized Trust Model for Assessment ofCloud Services,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 8(1), pp. 304-325, 2018.

[7] G. Somani, et al., “DDoS/EDoS attack in cloud: affecting everyone out there!” in *Proceedings of the 8th International Conference on Security of Information and Networks*, ACM, pp. 169-176, 2015.

[8] G. Somani, et al., “DDoS attacks in cloud computing: Issues, taxonomy, and future directions,” *Computer Communications*, vol. 107, pp. 30-48, 2017.

[9] D. Leggett, “CAPTCHAs tough on sales common way to test user tolerance,” Available: http://www.uxbooth.com/articles/captchas-tough-on-sales-common-way-to-test-user-tolerance/, 2009.

[10] V. S. M. Huang, et al., “A DDoS mitigation system with multi-stage detection and text-based turing testing in cloud computing,” in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on,* IEEE, pp. 655-662, 2013.

[11] C. S. Dule and Girijamma H. A., “Content an Insight to Security Paradigm for BigData on Cloud: Current Trend and Research,” *International Journal of Electrical and Computer Engineering (IJECE),* vol/issue: 7(5), pp. 2873-2882, 2017.

[12] B. Saini and G. Somani, “Index page based EDoS attacks in infrastructure cloud,” in *International Conference on Security in Computer Networks and Distributed Systems* (pp. 382-395). Springer, Berlin, Heidelberg, 2014.

[13] M. Masood, et al., “Edos armor: a cost effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments,” in *Multi Topic Conference (INMIC), 2013 16th International,* IEEE, pp. 37-42, 2013.

[14] J. Idziorek and M. Tannian, “Exploiting cloud utility models for profit and ruin,” in *Cloud Computing (CLOUD), 2011 IEEE International Conference on ,* IEEE, pp. 33-40, 2011.

[15] J. Idziorek, et al., “Attribution of fraudulent resource consumption in the cloud,” in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, IEEE, pp. 99-106, 2012.

[16] A. Chonka, et al., “Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks,” *Journal of Network and Computer Applications*, vol/issue: 34(4), pp. 1097-1107, 2011.

[17] T. Karnwal, et al., “A comber approach to protect cloud computing against XML DDoS and HTTP DDoS attack,” in *Electrical, Electronics and Computer Science (SCEECS), 2012 IEEE Students' Conference on,* IEEE, pp. 1-5, 2012.

[18] P. Negi, et al., “Enhanced CBF packet filtering method to detect DDoS attack in cloud computing environment,” *arXiv preprint arXiv:1304.7073*, 2013.

[19] S. Hao, et al., “A queue model to detect DDos attacks,” in *Collaborative Technologies and Systems, 2005. Proceedings of the 2005 International Symposium on,* IEEE, pp. 106-112, 2005.

[20] S. Jin and D. S. Yeung, “A covariance analysis model for DDoS attack detection,” in *Communications, 2004 IEEE International Conference on,* IEEE, vol. 4, pp. 1882-1886, 2004.

[21] N. Chandrakala and B. T. Rao, “Migration of Virtual Machine to improve the Security in Cloud Computing,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol/issue: 8(1), pp. 210-219, 2018.

[22] S. Zhao, et al., “Defend against denial of service attack with VMM,” in *Grid and Cooperative Computing, 2009. GCC'09. Eighth International Conference on,* IEEE, pp. 91-96, 2009.

[23] S. Yu, et al., “Can we beat DDoS attacks in clouds?” *IEEE Transactions on Parallel and Distributed Systems,* vol/issue: 25(9), pp. 2245-2254, 2014.

[24] B. Han, et al., “OverWatch: A Cross-Plane DDoS Attack Defense Framework with Collaborative Intelligence in SDN,” *Security and Communication Networks*, 2018.

[25] Z. Mašetić, et al., “SYN Flood Attack Detection in Cloud Computing using Support Vector Machine,” 2017.

[26] T. Mahjabin, et al., “A survey of distributed denial-of-service attack, prevention, and mitigation techniques,” *International Journal of Distributed Sensor Networks*, vol/issue: 13(12), 2017.

[27] W. J. Stewart, “Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling,” Princeton University Press, 2009.

[28] L. P. Slothouber, “A model of web server performance,” in *Proceedings of the Fifth International World Wide Web Conference,* 1996.

[29] N. Kulkarni, et al., “Real time control and monitoring of grid power systems using cloud computing,” *International Journal of Electrical and Computer Engineering (IJECE),* vol/issue: 9(2), pp. 941-949, 2019.

[30] S. Lagishetty, et al., “DMIPS-Defensive Mechanism against IP Spoofing,” in *Australasian Conference on Information Security and Privacy,* Springer, Berlin, Heidelberg, pp. 276-291, 2011.

[31] “Netwag Tool,” 2007. Available: http://ntwag.sourceforge.net/.

[32] P. Shamsolmoali, et al., “C_2DF: High Rate DDOS filtering method in Cloud Computing,” *Computer Network and Information Security no. August*, pp. 43-50, 2014.

[33] Kumaraswamy S. and M. K. Nair, “Bin packing algorithms for virtual machine placement incloud computing: a review,” *International Journal of Electrical and Computer Engineering (IJECE),* vol/issue: 9(1), pp. 512-524, 2019.

[34] T. Francis, “A Comparison of Cloud Execution Mechanisms Fog, Edge, and Clone Cloud Computing” *International Journal of Electrical and Computer Engineering (IJECE),* vol/issue: 8(6), pp. 4646-4653, 2018.

BIOGRAPHIES OF AUTHORS

A. Saravanan received his Master of Computer Applications Degree and M.Phil Degree from Bharathiar University, Coimbatore, India. Currently he is working as an Assistant Professor in the Department of MCA in Sri Krishna College of Technology, Coimbatore, India. He has 14 years of experience in teaching and 4 years in research. He has presented and published research papers in various international journals.

S. Sathya Bama received Master of Computer Applications Degree from Anna University, Chennai, India. Currently she is working as an Assistant Professor in the Department of MCA in Sri Krishna College of Technology, Coimbatore, India. She has 4 years of experience in both teaching and research. She has presented and published research papers in varies international journals & conferences.

Dr. Seifedine Kadry has a Bachelor degree in applied mathematics in 1999 from Lebanese University, MS degree in computation in 2002 from Reims University (France) and EPFL (Lausanne), PhD in applied statistics in 2007 from Blaise Pascal University (France), HDR degree in 2017 from Rouen University. At present his research focuses on education using technology, system prognostics, stochastic systems, and probability and reliability analysis. He is ABET program evaluator.

Lakshmana Kumar Ramasamy currently working as Assistant Professor cum Technical Trainer in Hindusthan College of Engineering and Technology, Coimbatore. Tamil Nadu. He is a global chapter Lead for MLCS [Machine Learning for Cyber Security]. Find him @ http://mlcsglobal.org/chapters-across-globe/. He is currently allied with company specific training of Infosys Campus Connect, Oracle WDP and Palo Alto networks. He did his UG in PSG group of Institutions and completed his PG in Kalasalingam University and PhD under Anna University, Chennai and his research is on semantic web services.