International Journal of Security and Its Applications
Vol.9, No.9 (2015), pp.197-212
http://dx.doi.org/10.14257/ijsia.2015.9.9.18

Chaotic Theory based Defensive Mechanism against Distributed Denial of Service Attack in Cloud Computing Environment

N. Ch. S. N. Iyengar[1] and Gopinath Ganapathy[2]

[1]*School of Computing Science and Engineering, VIT University, Vellore-632014, Tamil Nadu, India*
[2]*Director, Technology Park Bharathidasan University, Trichy-620023, India*
nchsniyengar48@gmail.com, gganapathy@gmail.com

Abstract

Cloud computing is an advantageous technology, which allows any enterprises to shift their data towards Cloud Service Provider (CSP) end. This shift poses an essential necessity for data being available all the time with a considerable level of security. Availability is an important concern for any subscribers as their sensitive data are prone to attack threats. Resource and data availability are most important security measure. So, blocking the attack traffic towards Data Center (DC) improves availability, but passive outwitting leads to high false positive and negative rate. This affects the legitimate requestors being outwitted. So, the proposed chaotic theory based defense mechanism considers the stability state of traffic and detects the anomaly traffic condition. The anomaly traffic condition is just the passive way of diminishing the effect of overload, but classifying them appropriately and allowing the non-attack case of overload improves the availability and utilization and reduces the false case rates. Considering several cases of overload threats and allowing the legitimate overload case improves efficiency. The simulation results proved that the mechanism proposed is deployable at an attack-prone DC for resource protection, which would eventually benefit the DC economically as well.

Keywords: *DDoS, Flash crowd, Cloud Computing, Chaos, Lyapunov's Stability*

1. Introduction

Cloud computing is a web technology, which serves required resources on-demand. Since the service requests are initiated and communicated via network layer sometimes through public network, they are vulnerable to getting lost or trashed by attackers. This creates the discrepancy in a service level agreement. Since the network traffic overload turns unpredictable, even though predicting the peak and off-peak traffic continued, but the Distributed Denial of Service (DDoS) still remains non-deterministic. This is because of the reason some group of hundreds of attackers distributed globally and launches the attack to subvert the DC. So, detecting such attack prone threats leads to resource protection and provisioning the protected resource to the requestors in need.

In a cloud computing environment, the data availability is maintained by deploying mirror servers which acts as a fault tolerant mechanism. But we proposed a protocol with an intention to protect DC against attackers and to maintain confidentiality with considerable data protection that resides in DC. Firstly, In order to improve data availability, the DCs should be protected against any anonymous attackers' entry. So, the incoming requesters are to be authenticated. In order to uniquely identify and validate users, we use lightweight encryption technique.

Cloud Flare experienced the largest DDoS attack flooding in February 2014, which is a record-breaking from 300 up to 400 Gbps attack. This attack is launched by anti-spammers Spamhaus, which was the largest DDoS attack up to date (2014) [1]. Some other notorious DDoS attack experiences are: Mafiaboy who succeeded in bringing down

ISSN: 1738-9976 IJSIA

the world's most popular websites, namely Yahoo, CNN, ebay, Dell and Amazon in February, 2000. Similar attacks were made towards the South Korean's largest newspaper, bank and United States forces created as a botnet over hundred thousands of computers in July, 2009 [2]. Still, several serious DDoS events lead to long outages to be identified and prevented.

The rest of the paper is organized as follows. Section 2 describes surviving techniques. Section 3 presents an overview of the proposed architecture and mechanism. Section 4 explains the working mechanism. Section 5 discusses the performance of the proposed mechanism. Section 6 deal with the advantages of the proposed approach and Section 7 provides the conclusions with an outline for our future work.

2. Surviving Techniques

Sporas, which is a "reputation" based methodology, has the reputation value of any user from 0 up to 3000 [3]. Obviously, the new user will have the minimal reputation value of 0. The current user's reputation will always be higher than the new users. Any two users should have only one rating value range from 0 up to 3000. In case the two users have several interactions, then only the recent value is considered. The evaluation is based on the more recent computed reputation value, because the more recent computed values results are current or much closer to the current behavior. The reputation value increases over the period upon good behavior and does not influence the initial low score.

Eschenauer proposed a framework for the evidence based trust management [4]. This considers the trust as a set of relationships between any two parties with the support of evidence. One way to generate evidence is through public-key cryptography. One of the entities in the network can create evidence for itself and for others. In order to create the evidence, the creator entity creates a piece of entity and signs it with the private key. It mentions its validity period and shares it to others with a public key for identification. The drawback here is that an entity could also revoke the shared evidence. Since the revoking option could allow any anonymous partner to create evidence and to revoke it, this would create chaos in the network.

Multilevel trust filtration [5] mechanism consists of four modular detection algorithms. Firstly, Link pre-fetch which attempts to identify the location of the incoming requestor. Secondly, the Requestor scrutinizes which verifies the network-specific data to authenticate the incoming requestor. Thirdly, Traffic data log which logs the request rate and request type and distinguishes the attack category, and eventual access after the right approval, decides the differential treatment for any different type of overload conditions and incoming traffic.

Unlike conventional DDoS detection and protection mechanisms, the Trilateral Trust mechanism [6] is a scheme which is extended in direction of detection with three sidelong functionalities. They are the preliminary traffic signaller, Authentic Trust launcher, historical trust Analyzer that scrutinizes several cases of requestors and continuously monitors their behavior and updates the trusteeship of the requestor towards the CSP.

A stochastic hill climbing approach [7] which is centralized load balancing mechanism and it is an incomplete approach for solving optimization problems. Genetic Algorithm [8] based load balancing proposes the three step operation, namely population generation, cross over to select the best fit resource configuration and mutation for choosing the probable best resource configuration. For insider threat detection, log management [9] has been proposed which uses log analysis with event correlation. Certain rule with a timestamp is continuously monitored using log watcher for better detection.

Network flow based entropy approach [10] analyzes requests per flow and improves detection accuracy by three steps. They are flow aggregation analysis based on traffic flow, fast entropy computation, and threshold refreshes for maintaining and improving the accuracy rate of detection. Application layer DDoS poses critical threats to web server

[11]. For such detection, the traffic has to be classified in order to avoid considering the flashcrowd as DDoS. This has been deployed with real-time frequency vector. Fuzzy logic based DDoS defense mechanism [12] considers the network packet parameters which attempts to detect the DDoS attack scenario, but there is a limitation or pitfall when the rate of traffic is huge as they process each traffic requests' packet parameters. DDoS Defense for web services in cloud discussion reveals the threat of XML attacks at the application layer [13]. Here the feature extraction and attack request model construction typically detects malicious requests.

DDoS defense mechanism [14] which uses a hop count filter, anomaly detectors, normal profile creation and attacker profile creation and comparing the incoming traffic to reduce false positive and false negative in order to improve the efficiency attacker detection schemes using Kullback-Liebler Divergence.

Intrusion Detection System (IDS) which enhances the system by distributing the IDS nodes across the network. Host IDS collects audit data from the operating system. Network IDS collect data from network packets. When any malicious intrusion is detected, the system generates reports and alerts. Fault tolerant workflow scheduling that makes use failure probability information. By combining the heuristic information of tasks and replicating the tasks helps to meet task deadline and save resources.

Sergio Nesmachnow; Santiago Iturriaga [15] proposed the solution for the scheduling the independent tasks in heterogeneous computing. Yona Raekow et al [16] proposed a license management scheme in a distributed environment which authenticates the users to access the remote servers. This proposed solution is compatible to all existing client-server architecture.

All the current surviving techniques either detect DDoS attack or create a secured architecture to protect the DC resource with a certain time lag. The proposed overload detection mechanism detects various kinds of attacks and authenticates them at different levels, and eventually makes the cloud environment free from attackers.

3. Overview of the Proposed Mechanism

3.1. Overview of Proposed Mechanism

DDoS and legitimate group follow a similar request pattern, but they differ in their traffic pattern. Because the intention of legitimates and DDoS are extremely different (i.e., Legitimates attempt to access DC resources, whereas DDoS group attempts to access illegal access or produce the blocking condition for legitimates which results in temporary outage or permanent shutdown of DC depending of the intensity of overload). It is common that the incoming traffic will be combination of legitimate and overload threats. But DDoS is not the only threat that affects DC Performance; there are other kinds of overload threats that attempts overload condition which has been discussed in detail in section 3.2.

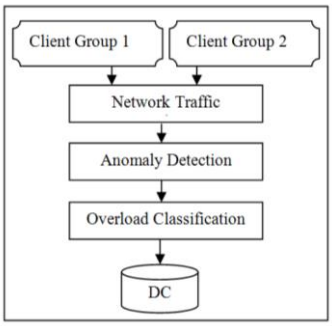

Figure 1. Overview of Proposed Mechanism

The proposed detection mechanism has three modules, namely Network traffic Analysis, Anomaly Detection and overload classification to validate the requestors' characteristics. The requestors' request traffic always bypasses Network Traffic Analysis phase where the state of stability is determined. The traffic stability state is derived based on the heuristics. The analyzed traffic is allowed to anomaly detection module which considers Lyapunov's stability theorem for the proposed dynamic traffic system, implies to concern the point at which the state of stability of solutions which is near to equilibrium. Based on the stability state and the probability measure of the uncertain traffic condition, the overload condition is predicted as normal or abnormal. Then the cause of overload is precisely predicted and the attack cases are filtered, non-attack cases are allowed to access DC resources.

3.2. Attack and Non-attack Cases

3.2.1. Attack Cases: Attack cases are the overload threats.

1. **Botnet:** Botnets are machines usually range from tens to thousands which continuously flood requests towards the target to subvert the victim target. The impact depends on the flooding rate and the number of machines joins the flooding rate.
2. **DDoS:** DDoS attacks are initiated and continued by some hundreds of attackers grouped and distributed globally who start populating the unwanted traffic packets with enormous size in order to acquire the memory, network resources and completely deplete them.
3. **Spoofers:** This is a kind of threat which is impersonation of legitimate requestor who attempts to join the traffic and attempt to acquire illegitimate access.
4. **Aggressive Legitimates:** This kind of threat is an internal or registered requestor attempting to overload, DC which often unnoticed, but contaminates CSP network resources if undetected pollutes memory and other resources of CSP.

3.2.2. Non-attack cases: Non-attack cases are either overload threat which does not affect DC or non-overload case.

1. **Flashcrowd:** Flash crowd is an overload condition caused by simultaneous incoming of the large number of legitimates but operated over a short period of time.
2. **Legitimates:** This group of requestors are innocent in traffic characteristic with the only intention to access DC resources get served by DC.

Incoming Traffic is fed through Anomaly detection module, irrespective of traffic behavior, because the periodical traffic behavior monitoring of each requestor acquires some information to predict the characteristic of each requestor which can be found in detail in section 4.2.

4. Working Mechanism of the Proposed Mechanism

The three phases of the proposed mechanism, namely network traffic analysis, anomaly detection and overload classification are discussed in detail below.

4.1. Network Traffic Analysis

The proposed mechanism considers Lyapunov's stability theorem because the incoming traffic keeps varying at various time period which remains uncertain as the incoming traffic vary in the rate at varied time period. In order to take the stable output into consideration for comparing the incoming traffic to inform accurate traffic deviation. By considering Lyapunov's stability theorem for the proposed

dynamic system, implies to concern the point at which the state of stability of solutions which is near to equilibrium [18].

Algorithm 1: Network Traffic Analysis
Input: Incoming Request Traffic
Output: Network Traffic Condition
Identify different states of traffic.
Obtain distinctive characteristic
Analyze the combinational traffic
Analyze the chaotic deviation
Apply and prevail preliminary traffic condition

Incoming traffic is sampled from the overall incoming traffic. The state of normal traffic is considered S_A. The state of abnormal traffic is considered S_B.

$$S_A + 1 = f(S_A) \quad (1)$$

$$S_B + 1 = f(S_B) \quad (2)$$

$f(S)$ is a nonlinear function which maps the state of incoming traffic.

From (1) and (2), the sequence of states can be generated as follows:

$$S_{A0}, S_{A1}, S_{A2}, ..., S_{AN} \quad (3)$$

$$S_{B0}, S_{B1}, S_{B2}, ..., S_{BN} \quad (4)$$

Equation (3), (4) represents the state of normal traffic behavior and abnormal traffic behavior.

$$S_{A0} + \Delta S_{A0}, S_{A1} + \Delta S_{A1}, S_{A2} + \Delta S_{A2}, ..., S_{AN} + \Delta S_{AN} \quad (5)$$

$$S_{B0} + \Delta S_{B0}, S_{B1} + \Delta S_{B1}, S_{B2} + \Delta S_{B2}, ..., S_{BN} + \Delta S_{BN} \quad (6)$$

Equation (5) represents the change in state of normal traffic behavior or surge normal traffic. Equation (6) represents a change in state of abnormal traffic behavior or surge abnormal traffic.

Considering legitimate or normal traffic S_{A0} and the overload or abnormal traffic $S_{B0} + S_{A0}$ is represented. Also, we presumed and resulted that at any period of time surge in non-attack case or normal or legitimate traffic settles back to normal with time $\Delta S_A(S_{A0}, t)$ which can be found in equation (5).

Chaotic deviation in normal traffic $\Delta S_A(S_{A0}, t)$ results in abnormal traffic = $\Delta S_B(S_{B0}, t)$ (7)

If normal traffic deviates at the time of ingress traffic and behaves chaotic, then it results in abnormal traffic behavior which is shown in equation (7).

Now the analyzed traffic is fed to the next level (i.e., Anomaly detection Phase) where the analyzed traffic is probed accurately to detect the traffic deviation.

4.2. Anomaly Detection

The key notion of differentiating the legitimate and overload threats is the traffic pattern. But considering only the traffic pattern does not scale down unnecessary traffic. Cutting down the complete overload threats needs some other factors to be considered for maintaining better resource protection.

Algorithm 2 : Anomaly Detection
Input: Analyzed Network Traffic
Output: Probed Network Traffic
Collect network traffic under no attack case

Compute training phase with network analysis
For any new incoming traffic compute as observing phase
Calculate Lyapunov's equation to predict the traffic state
Interchange observing phase as training phase
Detect abnormal traffic (attack case traffic)

By using Lyapunov's Exponent, the divergence between the state of stability (i.e., attack and normal traffic) can be described as T_{max}.

$$T_{max} = lim_{x\to\infty} \frac{1}{t} ln \frac{\Delta S(S_{A0}, t)}{S_{A0}} \quad (8)$$

If $T_{max} < 1$, then the deviation is caused by the non-attack traffic case entering into the system. Now the traffic is considered stable and non-chaotic. At this level, the network traffic is the cause of the legitimate clients The divergence caused may be because of non-attack case likely flashcrowd when deviates steep or legitimates when deviates gradually.

$T_{max} = 1$, then there is no deviation and the steady state is maintained. Now the incoming traffic is following the similar traffic pattern at present. This could be normal or abnormal traffic, which depends on the previous or training phase T_{max}.

$T_{max} > 1$, then the deviation is caused by the attack traffic case entering into the system. Now the network traffic is chaotic and unstable. This informs that $\Delta S_A(S_{A0}, t)$ deviates to $\Delta S_B(S_{B0}, t)$. This deviation can be a gradual deviation and also may be a surge deviation. Depending the on the deviation, the non-attack case is classified in later modules.

4.3. Overload Classification

Not only detecting the overload threats is highly essential, but also classifying them appropriately and deciding whether to redirecting them or dropping them is also an essential factor, because the overload threats are only caused by attack cases but also by non-attack cases.

Algorithm 3: Overload Classification
Input: Identified Overload Threat
Output: Classified Request Traffic
Collect pre-processed network traffic
Compute staged traffic analysis
Differentiate attack and non-attack cases
Consider non-attack case
Block abnormal overload traffic (attack case traffic)

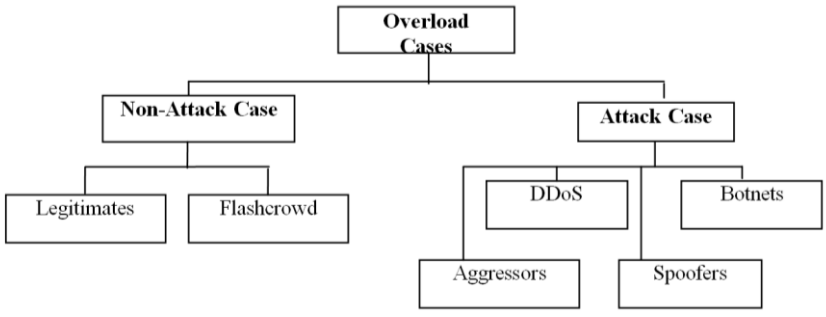

Figure 2. Overload Classification

5. Performance Evaluation

5.1. Experimentation

Jeyanthi and Iyengar [14] used OPNET as simulator to test the cloud computing environment. Jeyanthi *et al.* [19-20] experimented DDoS in cloud computing. We tested our proposed mechanism as a simulation experiment in OPNET Modeler. The experiments are performed in a campus network where DC requesters are grouped in fifteen subnets, and each subnet has got 100 workstations, 500 attackers and 1000 legitimate clients requesting for application-specific requests at each subnet. This way we created the attacker and legitimate profile and other devices, which would be needed to test our algorithm as an experiment. The traffic represents internet and the group of attackers is activated at varying time intervals. The attack profile is replicated to increase the attack strength to engage the DC resources like bandwidth, CPU and memory. The experiment is carried out with three different scenarios, namely the network with no attackers, networks with attackers and no detection mechanism in place, and finally the network with attackers.

5.2. Simulation Results

5.2.1. Network Traffic Condition: Network traffic condition is the rate at which the incoming traffic approach DC that includes the commingling attack case and non-attack case overload traffic rate. Figure 3 shows the overall traffic condition with and without the proposed mechanism.

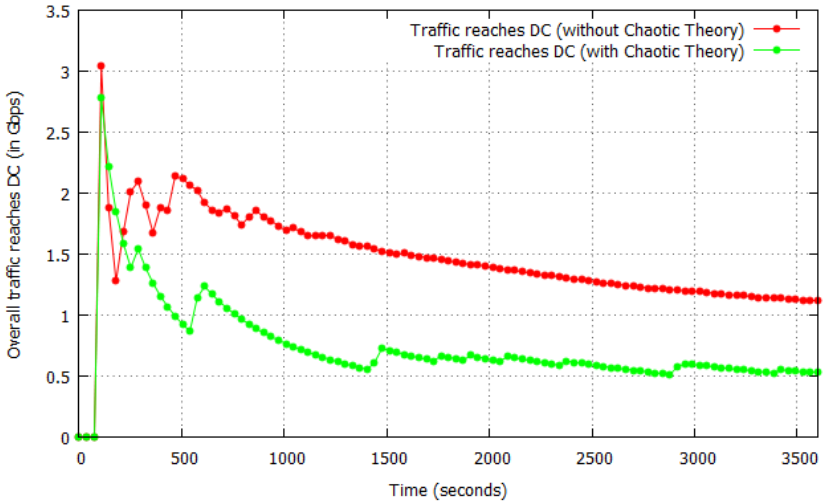

Figure 3. Network Traffic Condition

Table 1. Network Traffic Condition

Parameters	Minimum	Average	Maximum
Traffic at DC with Chaotic theory	**0**	0.746	30
Traffic at DC without Chaotic theory	**0**	1.496	27

Network Traffic approaching DC = $\sum_{i=0}^{n}(\sum_{j=0}^{m} T_{ij})$ (9)

where, n = number of requestors; m = overall of traffic produced by each requestors.

Initially, both scenarios in figure 4 behave similarly (i.e., at 2nd to 3rd minute), the traffic increases steeply. But when time moves on, the proposed mechanism shows the tiny oscillations which are due to outwitting attackers and joining again after session timeout which can be observed. Whereas the traffic rate without proposed mechanism shows the greater divergence of traffic, which is because of the random attacker activation and remained constant after some period of time and not dropping further. Also the traffic reaching DC can be reduced 50% can also be observed in figure 3.

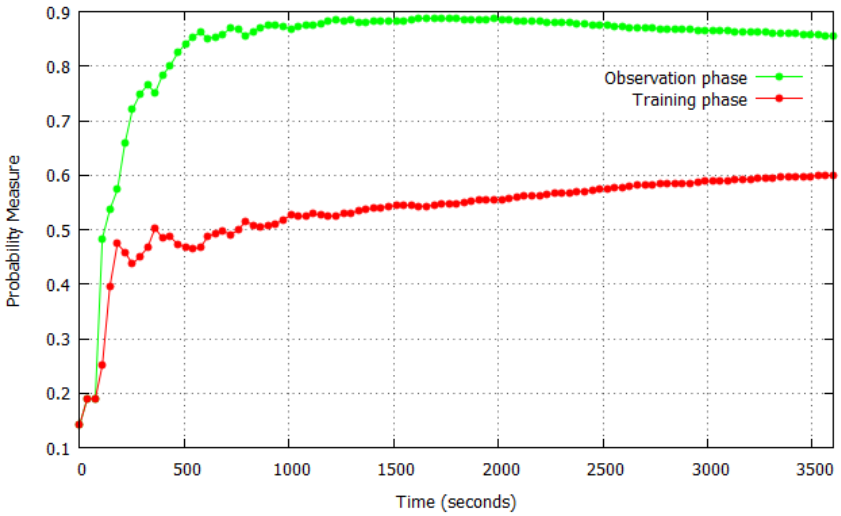

Figure 4. Chaotic Measurement

5.2.2. Chaotic Measurement: Chaotic measurement is a measure of traffic stability from the heuristic data. Figure 4 shows the traffic behavior profiling with the proposed mechanism.

Table 2. Chaotic Measurement

Parameters (with Chaotic Theory)	Minimum	Average	Maximum
Training Traffic Profile	0.14	0.531	.601
Observed Traffic Profile	0.14	0.832	0.888

The chaotic or state of confused traffic is measured based on profiling the incoming requestors' traffic rates.

5.2.3. False Case Rate: False case rate is a measure of traffic deviation analysis from the regular traffic probing Figure 5 shows the overall false case rates (false positive and false negative) scenario for the proposed mechanism.

Table 3. False Case Rate

Parameters (with Chaotic Theory)	Minimum	Average	Maximum
False Case Rate	0.02454	0.011791	0.01265

$$\text{False case rate} = \frac{LT}{TA} + \frac{AT}{TL} \quad (10)$$

Where LT = legtimate traffic requests; TA = Total number of attack requests; AT = Attack requests; TL = Total number of legitimate requests.

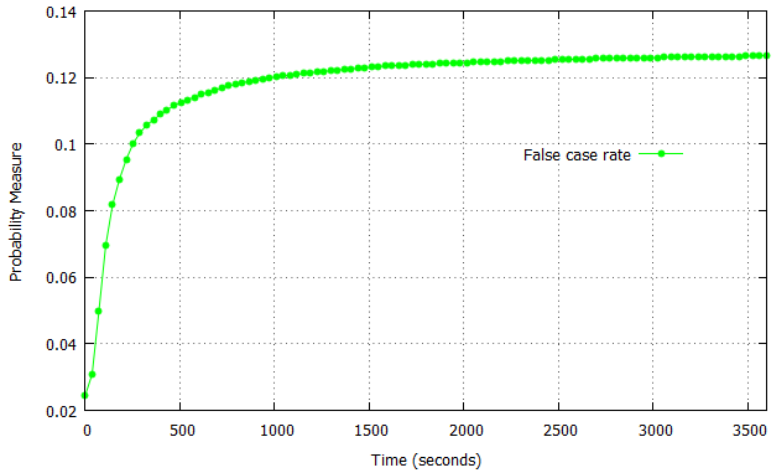

Figure 5. False Case Rate

False case rate analyses the strength of the proposed mechanism, here the false case rate is the combined rate of false positives and false negatives. The probable measure of the false case rate shows that on an average the detection is 89% (approx) accurate, as 11% of them found false case rates can be seen in figure 5.

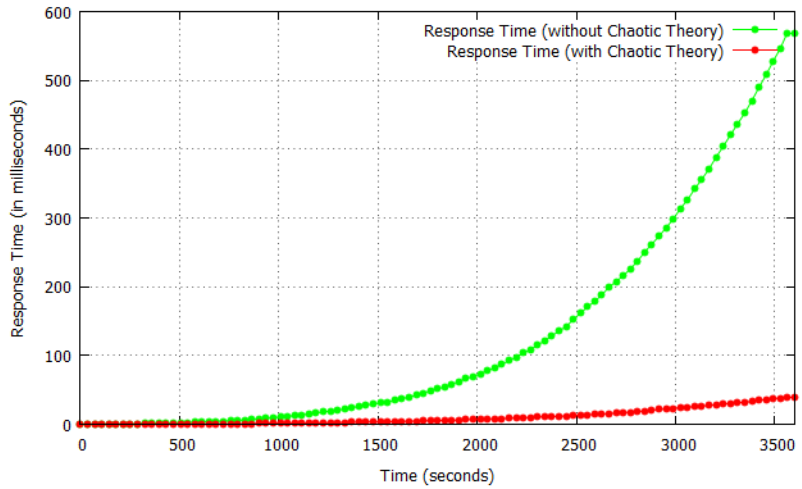

Figure 6. DC Processing Performance

Table 4. DC Processing Performance

Parameters	Minimum	Average	Maximum
DC performance with Chaotic theory	**0**	9.976622	39
DC performance without Chaotic theory	**0**	126.2108	568

5.2.4. DC Processing Performance: DC processing Performance is a measure of time to process the received request. This excludes the load balancing activity and request

queueing. Figure 6 shows the DC performance scenarios with and without the proposed mechanism.

Figure 6 shows that our proposed mechanism has better performance in processing the incoming requests because the incoming traffic is not processed directly instead they are probed and classified based on the traffic behavior. As the incoming traffic would be very less after the classification, the performance seems better which in turn improves response time and reduces the queueing delay.

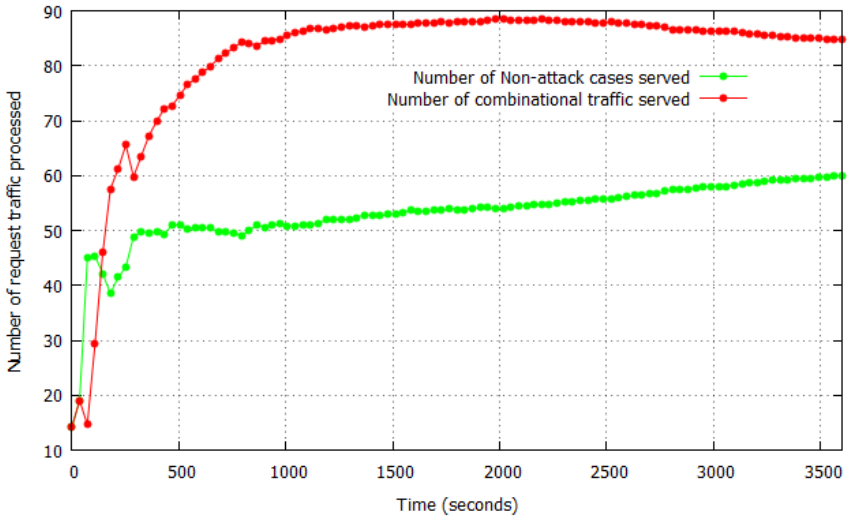

Figure 7. Goodput

5.2.5. GoodPut: Goodput is the application level throughput, which includes only the number of non attack case overloads served by ignoring the attack case overload threats. Figure 7 shows the goodput with (Non-attack traffic served) and without (combinational, i.e., attack and non-attack traffic served) the proposed mechanism.

Table 5. Goodput

Parameters	Minimum	Average	Maximum
Goodput with Chaotic theory	14	5.28	88
Goodput without Chaotic theory	14	8.12131	59

$$\text{Goodput} = \sum RT - \sum RA \tag{11}$$

Where RT = overall request traffic; RA = overall attack-case traffic.

Figure 7 shows the number of requestors served without the proposed mechanism (i.e., legtimates, flashcrowd, Aggressors, spoofers) and the number of requestors served with the proposed mechanism (legitmates and flashcrowd). Upon detecting the overload threats they are classified appropriately and only the non-attack case traffic is allowed to access DC. Whereas without our proposed mechanism several attack cases are not considered and treated as legitimates (i.e, spoofers, Aggressors) which leads to improved goodput which is actually not.

5.2.5. Traffic Pattern Recognition: Traffic pattern recognition is the part of Network analysis and anomaly detection module for monitoring the incoming traffic behavioral analysis of the overall traffic approaching DC. Figure 8 shows the traffic patterns with attack and non-attack cases.

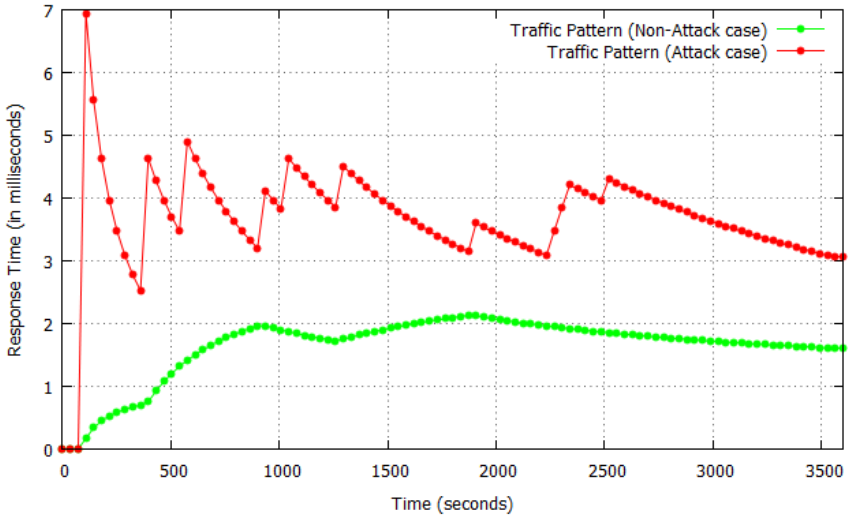

Figure 8. Traffic Pattern Recognition

Table 6. Traffic Pattern Recognition

Parameters (with Chaotic Theory)	Minimum	Average	Maximum
Traffic Pattern (Attack Case)	0	3.671	6.9
Traffic Pattern (Non - Attack Case)	0	1.635	2.1

The traffic pattern is the characteristic which can be analyzed to predict the overload condition. So, the proposed mechanism treats the incoming traffic as attack case and non-attack case. So, the non-attack case (legitimates, flashcrowd) behaves different in their traffic pattern. While attack cases (Bots, spoofers, aggressors, DDoS) behaves different in their traffic pattern. The reason of sharp oscillation is because of the detection of misbehaved requestors. The exponential traffic growth is controlled, but the outwitted requestors join the network after their session expiry. While irksome oscillation is change in legitimate traffic arrival rate at varied interval of time can be seen in figure 8.

5.2.6. Request Queue based Bandwidth Accommodation: Request Queue is a measure of recognizing traffic overload condition. Figure 9 shows the number of legitimate requests queuing scenarios with and without the proposed mechanism.

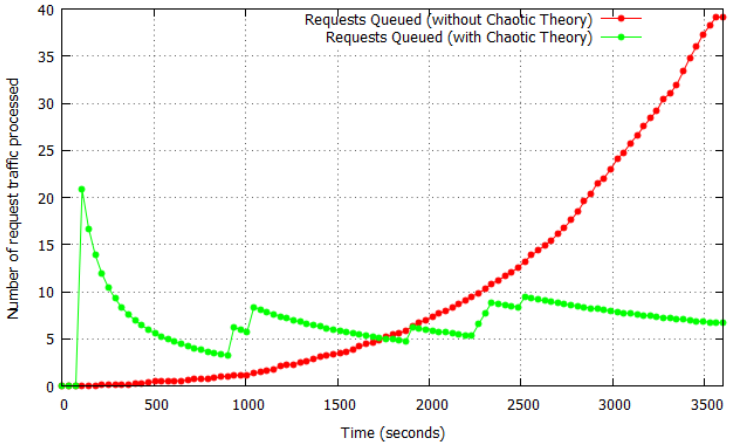

Figure 9. Request Queued at DC

Table 7. Request Queued at DC

Parameters	Minimum	Average	Maximum
Request Queued with Chaotic theory	0	5.07	20
Request Queued without Chaotic theory	0	9.97	39

From figure 9, the requests queued for processing grows exponentially because of the combined traffic sources. If they are classified before they reach DC (i.e., overload classification phase) the request queueing can be controlled or eliminated trivially with high-end DCs. The request queueing Is the cause of poor response time and delayed processing. Request queueing has been brought under control can be seen in figure 9. The oscillations are the cause of change in traffic behavior and outwitting the misbehaved clients.

6. Advantages of Proposed Mechanism

Efficient Analytic scheme: The proposed mechanism proves better by considering several cases of overload instead of classic rate limitation. Moreover the network traffic analysis phase provides some traffic information which acts as an early alarm system for optimized DC operation.

Predicting uncertain overload: The chaotic scheme preliminarily considers the different states of stability and consistently informs the change in state of traffic stability which helps in predicting the uncertain traffic overload cases.

Structured levels of detection: The continuous traffic monitoring, processing and classifying is the protocol for any incoming requestor. So, even with the normal traffic condition, the incoming traffic continuously monitored and departmentalized for classifying them. This moduled or structured mechanism reduces the overload threats reaching DC.

Highly sensitive detection: The proposed traffic behavioral analysis improves accuracy and also identifies the overload threats which lead to lower false positives and false negatives. Conversely the low rate attack would fail with the predefined threshold predictive schemes.

Table 8. QoS Parameters Observations

QoS Parameters	Non attack cases	Attack cases
Bandwidth utilization	450 kb/sec (on average)	5.4 Mb/sec (on average)
DC memory overhead	No	Exponential increased
Jitter	Not Found	Drastically Found
Packet loss	0%	56% (approx)
Packet Delivery Fraction	98% legitimates	47% legitimates

Table 8 shows a complete examination of the network QoS Parameters considered while conducting the experiment with the Lyapunov chaotic theory along with enhanced anomaly detection and classification which led the cloud resource to reserve from the attackers and to provision them on demand.

6.1. Profit Analysis

Profit Analysis is the cost incurred in processing and provisioning service to the intended requestors which includes the network resources, memory resources, storage resources. Computed cost is the cost calculated based on the resources used at CSP end.

Let N = time in hours; CI_{BW} = Bandwidth cost; CI_{MEM} = RAM cost of each physical equipment; CI_{VM} = VM cost of each physical equipment and CI_{DS} = Data stored within a DC.

Total cost incurred at $DC = \sum_{i=1}^{N} \{ CIBW + CIMEM + CIVM + CIDS \}$ (12)

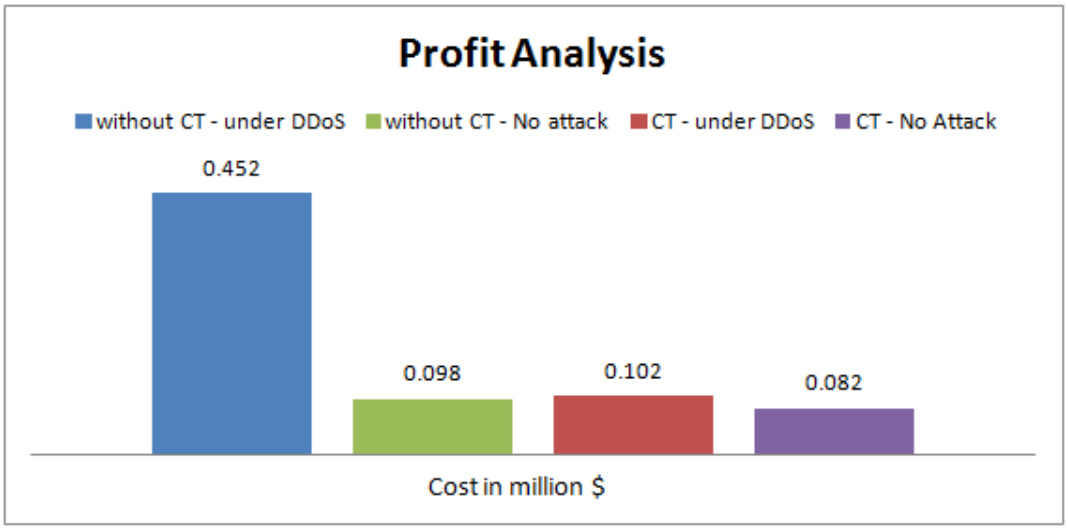

Figure 10. Profit Analysis

The costs used are $ 0.1/Gb for any data transmission at the DC and $ 0.05/second for any memory resident operations at the DC. Figure 10 shows the overload and non-overload scenarios with and without proposed mechanism. The proposed mechanism has shown the cost efficiency, which in turn shows the better resource protection and also results subscribers' sensitive data protection against the overload cases considered. Resource protection results in improved availability for the intended subscribers.

7. Conclusion and Future Work

This paper considers the different types of overload and classifies them as attack cases and non-attack cases which not only distinguishes DDoS and legitimates but also considers several other cases which when filtered improves the network performance and improves the DC resource reservation for better utility. Various simulation scenarios proves the proposed chaotic based moduled defense mechanism improves CSP performance and efficiency.

Our future work is to enhance the proposed mechanism to improve detection efficacy and to apply still better analytic scheme to detect the low-rate DDoS attack to improve the QoS observation.

References

[1] https://www.cloudflare.com/under-attack (accessed on December **2014**).
[2] http://www.thedailybeast.com/articles/2010/12/11/hackers-10-most-famous-attacks-wormsand-ddos- takedowns.html (accessed on December **2014**).
[3] G. Zacharia, 'Trust Management through Reputation Mechanisms', Workshop in Deception, Fraud and Trust in Agent Societies, Third International Conference on Autonomous Agents (Agents'99), ACM, **(1999)**.

[4] L. Eschenauer, V. D. Gligor and J. Bara, 'On Trust Establishment in Mobile Ad Hoc Networks', Security Protocols Springer, **(2004)**, pp. 47-66.
[5] N.Ch. S.N. Iyengar, Gopinath Ganapathy, P.C. Mogan Kumar, and Ajith Abraham, 'A multilevel thrust filtration defending mechanism against DDoS attacks in cloud computing environment', International Journal of Grid and Utility Computing, vol. 5, no. 4, **(2014)**, pp. 236-248.
[6] N. Ch. S. N. Iyengar and G. Ganapathy, 'Trilateral Trust Based Defense Mechanism against DDoS Attacks in Cloud Computing Environment', Cybernetics and Information Technologies, vol. 15, **(2015)**, no. 2, pp. 119-140.
[7] B. Mondal, K. Dasgupta and P. Dutta, 'Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach', Procedia Technology, vol. 4, **(2012)**, pp. 783-789.
[8] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal and S. Dam, 'A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing', Procedia Technology, vol. 10, **(2013)**, pp. 340-347.
[9] A. Ambre and N. Shekokar, 'Insider Threat Detection Using Log Analysis and Event Correlation', Procedia Computer Science, vol. 45, **(2015)**, pp. 436-445.
[10] J. David, Ciza Thomas, 'DDoS Attack Detection Using Fast Entropy Approach on Flow-Based Network Traffic', Procedia Computer Science, **(2015)**, Vol. 50, pp. 30-36.
[11] Wei Zhou, Weijia Jia, Sheng Wen, Yang Xiang, Wanlei Zhou, 'Detection and defense of application-layer DDoS attacks in backbone web traffic', Future Generation Computer Systems, **(2014)**, Vol. 38, pp. 36-46.
[12] N.Ch.S.N. Iyengar, Arindam Banerjee, Gopinath Ganapathy, 'A Fuzzy Logic based Defense Mechanism against Distributed Denial of Service Attack in Cloud Computing Environment', International Journal of Communication Networks and Information Security (IJCNIS), **(2014)**, Vol.6, No.3, pp.233-245.
[13] Thomas Vissers, Thamarai Selvi Somasundaram, Luc Pieters, Kannan Govindarajan, Peter Hellinckx, 'DDoS defense system for web services in a cloud environment', Future Generation Computer Systems, **(2014)**, Vol.37, pp. 37-45.
[14] Jeyanthi N, Iyengar N.Ch.S.N, Mogan Kumar P.C, Kannammal A, 'An enhanced entropy approach to detect and prevent DDoS in cloud environment', International Journal of Communication Networks and Information Security, **(2013)**, Vol. 5, No. 2, pp.110–119.
[15] P. Varalakshmi, S. Thamarai Selvi, 'Thwarting DDoS attacks in grid using information Divergence', Future Generation Computer Systems, **(2013)**, Vol.29, No.1, pp. 429–441.
[16] Sergio Nesmachnow, Santiago Iturriaga, 'Multiobjective grid scheduling using a domain decomposition based parallel micro evolutionary algorithm', International Journal of Grid and Utility Computing, **(2013)**, Vol.4, No.1, pp.70 – 84.
[17] Yona Raekow, Christian Simmendinger, Domenic Jenz, Piotr Grabowski, 'On-demand software licence provisioning in grid and cloud computing', **(2013)**, International Journal of Grid and Utility Computing, Vol.4, No.1, pp.10 – 20.
[18] Ashley Chonka, Jaipal Singh, Wanlei Zhou, 'Anomaly Detection of Distributed Denial of Service Attacks by Non-Liner Dynamics', IEEE COMMUNICATIONS LETTERS, **(2009)**, January.
[19] N. Ch. S. N. Iyengar, Gopinath Ganapathy, 'An Effective Layered Load Balancing Mechanism to defense against DDoS in Cloud Computing Environment', International Journal of Security and its Applications, **(2015)**, Vol. 9, No. 7, pp.17–36.
[20] Jeyanthi N, Iyengar N.Ch.S.N, 'Packet resonance strategy: a spoof attack detection and prevention mechanism in cloud computing environment', International Journal of Communication Networks and Information Security, **(2012)**, Vol. 4, No. 3, pp.163–173.

Authors

N. Ch. S. N. Iyengar (1961) he is currently a Senior Professor at the School of Computing Science and Engineering , VIT University, Vellore-632014, Tamil Nadu, India .He had 30 yrs of teaching experience. His research interests include Agent-Based Distributed secure Computing, Intelligent Computing, Network Security, Cloud Computing and Fluid Mechanics. He has authored several textbooks and had nearly 162 research publications in reputed peer reviewed International Journals. He delivered many keynote /invited lectures and served as PCM//TCM/reviewer for many *Int'l Con,s.*. He is Editor in Chief for *Int'l J.* of Software Engg. & Appl., (IJSEA) of AIRCC, Guest Editor for *SI* on Cloud Computing and Services of

Int'l J. of CNSS, and EB member for *Int'l J's* like IJAST of SERSC, IJConvC of Inder Science and many more.

Gopinath Ganapathy, PhD and is the Professor and Head of the Dept. of Computer Science and Engineering, Bharathidasan University, India. He did his under graduation and post graduation in 1986 and 1988 respectively from Bharathidasan University, India. He obtained his PhD degree, in Computer Science in 1996, from Madurai Kamaraj University, India. Received Young Scientist Fellow Award for the year 1994 and eventually did the research work at IIT Madras. He published around 60 papers. He is a member of IEEE, ACM, CSI, and ISTE. He was a Consultant for 10 years in the international firms in the USA and the UK, including IBM, Lucent Technologies (Bell Labs) and Toyota. His research interests include Modeling, Patterns, NLP, Web Engineering, and Text Mining.