Developing a Replication Strategy Based on Fuzzy Logic to Enhance Data Availability in Cloud Data Center

Jeyaraman Sathiamoorthy [1], M. Usha [2] and P. Senthilraja [3]
[1] Professor, Department of Computer science and Design, RMK Engineering College (Autonomous), RSM Nagar, Kavarapettai, Gummidipoondi Taulk, Thiruvallur District-601206, India.
[2] Professor and Director, Department of Master of Computer Application, MEASI Institute of Information Technology, Royapettah, Chennai 600014, India.
[3] Research Scholar, Anna University, Chennai 600025 India
*Corresponding e-mail: senthilrajamtech@gmail.com

Abstract- In massive cloud systems, efficient data management is crucial. This element's management is greatly streamlined by file replication. The purpose of this project is to spread replicates over a variety of data centers in order to boost data availability in cloud data centers. How clones are created could depend on how useful and accessible the data is in the cloud data center. Throughout the replication process, high-access data is given precedence. The process of choosing the best copy to deploy across many data centers is known as replica selection.

Index Terms— **Computing in the cloud, data centres, and replication are examples of index terms**

I. INTRODUCTION

The users the flexibility of accessing a shared resource pool from any location globally, whenever needed, without concerning themselves with system installation or maintenance. Provisioning hosted services for users is a key aspect for ensuring high data availability. The proposed approach encompasses three primary categories. The first section entails replica creation, contingent upon the data's accessibility. Subsequently, the second step involves selecting the most suitable copy based on network performance. The third section focuses on the replication installation process and replacement of older replicas when necessary. The final stage integrates the replica selection mechanism, utilizing fuzzy logic to determine the optimal quantity of replicas and enhance data availability through this replication approach.

II. REPLICATION OF DATA

Data replication stands as a viable approach to meet data integration needs. Through replicating data, synchronization is maintained, ensuring data availability, promoting efficient data expansion, and leveraging the latest data to enhance revenue. Real-time analytical data synchronization adds value to a diverse range of businesses.

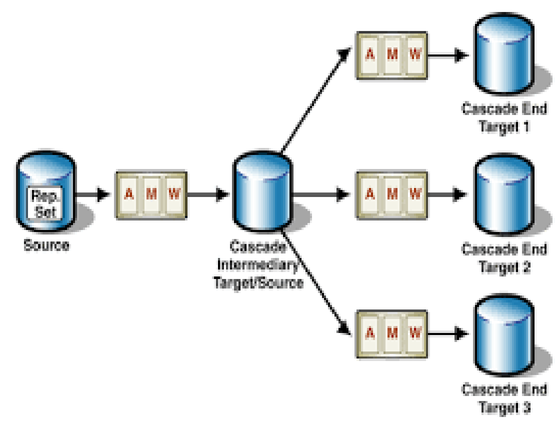


Fig. 1 Statistics for Duplication

Data replication, owing to its robust capabilities in transferring vast data volumes swiftly with minimal latency, proves highly effective in distributing workloads across multiple sites and ensuring continuous data accessibility. 1Illustrated in Figure 1 is the fundamental methodology was employed for data replication.

The concept of data replication involves maintaining replicas, essentially multiple copies of the data. Replication enhances data availability, enabling data access even if some replicas are inaccessible. Moreover, it optimizes system performance by reducing latency for users, redirecting them to use local copies of data instead of accessing remote networks.

WORKS THAT ARE INTERCRETED

Cloud computing grapples with notable challenges concerning the effective implementation of data management strategies. Research has delved into the realm of replication, classifying the processes into two primary categories: While dynamic replication can dynamically generate or remove clones based on data accessibility. In 2012, Bakhta Meroufel and team introduced a dynamic replication strategy for hierarchical grids, considering system crashes and failures [1]. The bedrock of dynamic replication lies in data accessibility and its usage prevalence. Depending on data popularity, the proportion of available data might increase if the data is highly favored.

DHR deletes files within the local area network (LAN) or files with short transfer times. The algorithm stores replicas in the most accessed location, diverging from the approach of storing originals in various locations. In 2015, Reena S. More and team introduced the custom partitioning algorithm [7]. The custom partitioning method aims to break down extensive problems into more manageable subproblems. In this suggested approach, data would be replicated on a high-performance node, reducing energy consumption during data processing [8].

WORK TO BE PROPOSED

The main objectives of this endeavor include augmenting data availability through distributed replication across diverse Information hubs and curbing dynamism ingesting through regulating. Our strategy based on fuzzy logic led to notable reductions in energy consumption. We employ a technique known as MORM

A. *System Architecture*

Four modules make up the proposed architecture.

1. One of the modules is data centre configuration.

2. Making copies

3. Replica choice

4. Placing a duplicate

Configuration of the Data Center, to start

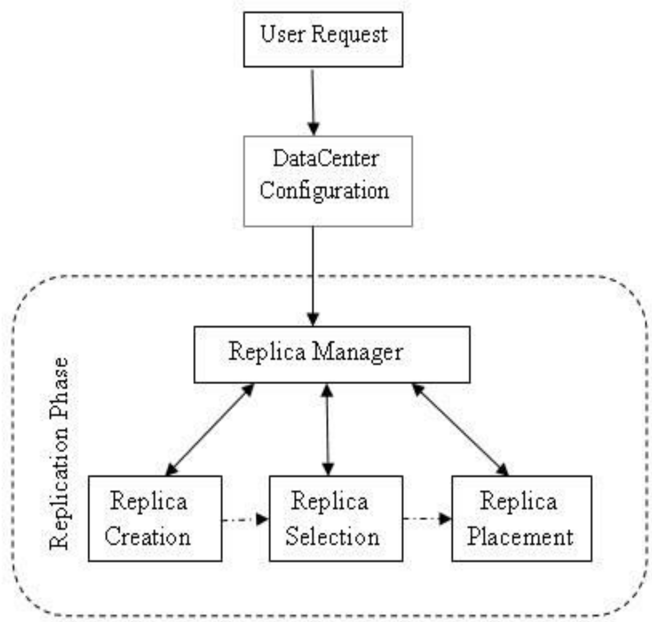


Fig. 2. Schematic of the proposed system

Replica Manager, Level Two: The replica manager oversees all replication phase functions, encompassing replica development, replica selection, and replica placement. This involves information on replica specifics such as their location, size, and quantity. If the primary data center experiences an outage, the replica manager redirects user requests to an operational zone. a) Replica Development: Replication occurs during the replica construction phase, with the timing influenced by user request numbers. The process involves generating the required additional copies and distributing them across different data centers.

Typically limit active replication copies to three. Once a replica is successfully installed in the primary data center, users gain immediate access to the file. Condition there's sufficient storage space in the primary data center, a replica is constructed and stored there. b) Replica Selection: Selecting a replica involves choosing the data center with the best replica from all available data centers (e.g., primary and secondary zones).

This optimizes provisioning efficiency when needed, reducing strain on the network bandwidth. Replica selection optimizes data processing in cloud data centers by providing the most suitable data set for user requests, resulting in minimal memory usage and swift data processing.

Two considerations for replica selection include: Opt for a nearby data center replica to reduce bandwidth requirements. Choose a network region with fewer user requests to minimize distributing replicas of the original resource across multiple data centers. The ongoing process of relocating replicas to new data centers without the resource reduces storage space requirements, and replica updates continue on a regular basis.

5. APPARATUS EXAMINATION

Mist Predictor: An imperative necessity to adopts a graphical user interface, simplifying the evaluation of social networking tools concerning the geographical dispersion. Dynamic re organization The design of the Cloud Analyst tool is depicted in Figure 3, with different colors representing diverse regions. In this representation, six geographical areas are encompassed.

Each region hosts one or more user bases linked to data centers situated in undisclosed regions.

TABLE I. Correlation of the three possibilities for 10 Under and 1 DC

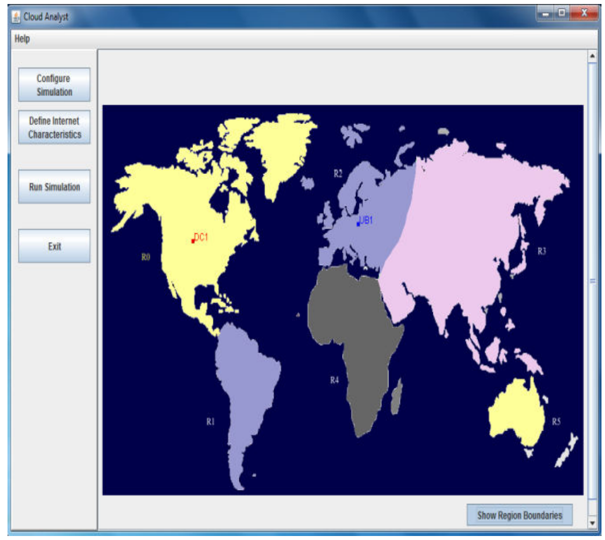
Scenario 2: 50 User Bases & 1 Data Center

Data center region : 3
No.of data centers : 1
No.of userbase : 50

A) Assessment of Service Broker Policies: Comparative analysis across three scenarios.

Information centre area- 3 Data centre count: One User count: 10

Fig. 3. Cloud analyst screenshot



	Closest data center	Optimize response time	Reconfigure dynamically
Overall response time	Avg (ms) - 571.23 Min (ms) - 38.86 Max (ms) -1245.12	Avg (ms) - 570.21 Min (ms) - 39.36 Max (ms) - 1260.12	Avg (ms) - 571.70 Min (ms) - 39.11 Max (ms) - 1245.61
Data center processing time	Avg (ms) - 0.27 Min (ms) - 0.02 Max (ms) - 0.86	Avg (ms) - 0.27 Min (ms) - 0.01 Max (ms) - 0.87	Avg (ms) - 0.75 Min (ms) - 0.02 Max (ms) – 30.01
Data center request servicing times	Avg (ms) - 0.27 Min (ms) - 0.02 Max (ms) - 0.86	Avg (ms) - 0.27 Min (ms) - 0.01 Max (ms) - 0.87	Avg (ms) - 0.75 Min (ms) - 0.02 Max (ms) – 30.01
Total virtual machine cost ($)	0.50	0.50	3.26
Total data transfer cost ($):	0.64	0.64	0.64
Grand total: ($)	1.14	1.14	3.90

	Closest data center	Optimize response time	Reconfigure dynamically
Overall response time	Avg (ms) - 647.14 Min (ms) - 37.79 Max (ms) - 1300.12	Avg (ms) - 647.48 Min (ms) - 38.61 Max (ms) - 1305.11	Avg (ms) - 647.78 Min (ms) – 38.41 Max (ms) - 2680.01
Data center processing time	Avg (ms) - 0.25 Min (ms) - 0.01 Max (ms) - 0.90	Avg (ms) - 0.25 Min (ms) - 0.01 Max (ms) - 0.90	Avg (ms) - 0.86 Min (ms) - 0.01 Max (ms) – 1635.01
Data center request servicing times	Avg (ms) - 0.25 Min (ms) - 0.01 Max (ms) - 0.90	Avg (ms) - 0.25 Min (ms) - 0.01 Max (ms) - 0.90	Avg (ms) - 0.86 Min (ms) - 0.01 Max (ms) – 1635.01
Total virtual machine cost ($)	0.50	0.50	3.25
Total data transfer cost ($):	3.20	3.20	3.20
Grand total: ($)	3.70	3.70	6.46

TABLE II. Correlation of 50 UB and 1 DC under three different circumstances

Considering response time, processing duration, data transfer expenditures, and related aspects, demonstrate nearly identical performance. In performance and cost, these two policies outshine the dynamically reorganize policy.

VI. ANALYSIS OF ALGORITHM

Algorithm- Placing replicas

```
Input: Integer r, the number of replicas
Input: Integer dc, the number of datacenters

Output: Replicas [0...dc - 1][0...r -1]

for all i such that i = 0 or i < 3 do
//(i – maximum allowable count of replicas. Where i=3
{
      Preload node dc's replicas with dc
}
for all i such that i < 3 do
{
      repeat until i=3
      z = random node, s.t. 0 = z < dc
      v = Rep[z]
      until z = i
      if v = i and Rep[i] = z then
      {
            valid rep = 1
            if Rep[i]== v or Rep[i]== Rep[z] then
            {
                  valid rep = 0
            }
            if valid rep then
            {
                  Rep[z]= Rep[i]
                  Rep[i] = v
```

The previously mentioned algorithm also includes an outline of the replication placement procedure. In this context, inputs encompass the replica count and the data center count, producing an output of successfully replicated replicas across the designated three for a specific resource, initiating the replication process requires preloading the original copy. The recently created replica must be situated on node Z, assumed to be a random node, and the replica count should align with the intended number of data centers containing the resource. This process continues until reaching the count of 3. If a replica does not precisely match the resource it was copied from, it is considered invalid (i.e., valid_rep=0). Conversely, if the replica perfectly matches the resource, it is considered valid (i.e., valid_rep=1).

ISSN (ONLINE):2456-5717 Vol.11, Issue.3, March 2025

VII. IMPLEMENTATION RESULTS

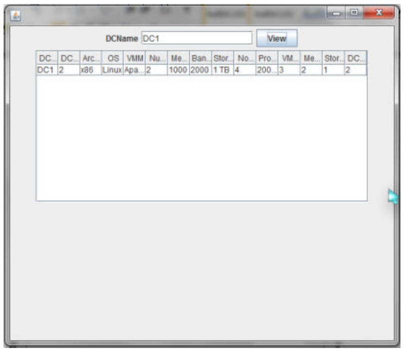


Figure 4 illustrates a screen capture displaying DC details.

The information that was retrieved about the chosen data center is shown in Figure 4. Since these features are only available to cloud service providers, only they have the power to add or alter data in the cloud.

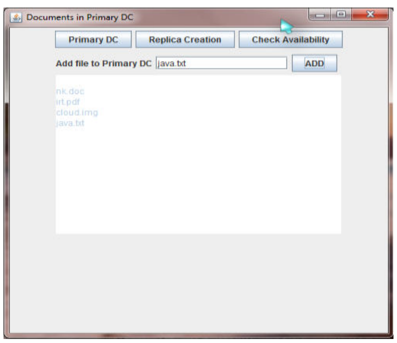


Figure 5 provides a screen capture showcasing DC documents.

Figure 5 displays the documents that may be found in the main data center and are controlled by users under the supervision of the service provider.

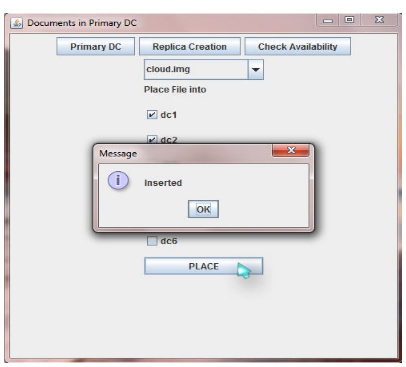


Figure 6 presents a screen capture of the replica creation process.

Figure 6 shows how to make copies in practice. The selected file ('cloud.img') is stored in two different data centers, DC1 and DC2, respectively.

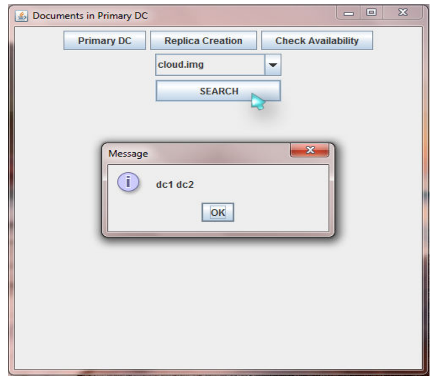


Figure 7 displays a screen capture for availability verification.

The verification of Figure 7, where both Data Center 1 and Data Center 2 contain copies of the file named 'cloud.img.' The earlier described implementation integrates modules for responsibility of the service provider. The process of duplicating a file stored in the cloud, based on the file's access frequency, is denoted as replica creation.

VIII. IMPLEMENTATION IN TOOL

(CLOUD ANALYST)

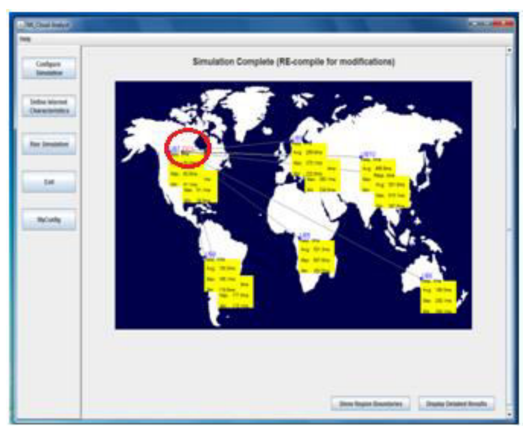
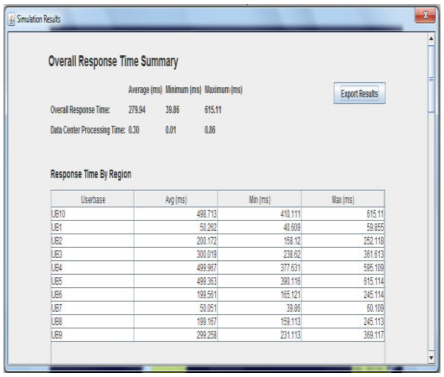


Figure 8: Simulation employing a Single Data Center & 10 User Bases.

Picture 8 portrays simulation outcomes utilizing a single information Hubs and ten user bases. In this scenario, due to the presence of a solitary data center, the concept of replication is unachievable.



Simulation Results

Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	279.94	39.86	615.11
Data Center Processing Time:	0.30	0.01	0.86

Export Results

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB10	498.713	410.111	615.11
UB1	50.262	40.609	59.855
UB2	200.172	158.12	252.118
UB3	300.019	238.62	361.613
UB4	499.967	377.631	595.109
UB5	499.363	390.116	615.114
UB6	199.561	165.121	245.114
UB7	50.051	39.86	60.109
UB8	199.167	159.113	245.113
UB9	299.258	231.113	369.117

Figure 9: Summary of Overall Response Time (For Figure 8.1).

The overall response time is shown in Figure 9 along with response times for various user bases. Response time is used in this situation to measure the algorithm.

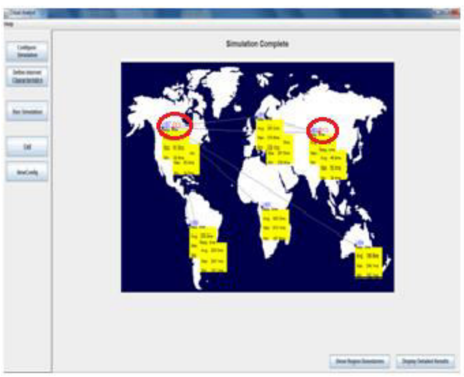
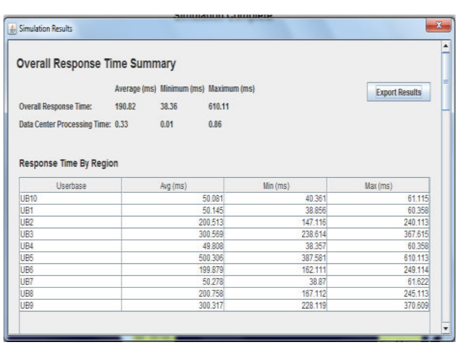


Figure 10: Simulation utilizing 2 Data Centers & 10 User Bases.

Figure 10 showcases simulation results employing two data centers and 10 user bases, enabling the replication concept. As a result, the same resource can be held by two different data centers.



Simulation Results

Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	190.82	38.36	610.11
Data Center Processing Time:	0.33	0.01	0.86

Export Results

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB10	50.081	40.361	61.115
UB1	50.145	38.856	60.358
UB2	200.513	147.116	240.113
UB3	300.569	238.614	367.615
UB4	49.808	38.357	60.358
UB5	500.306	387.581	610.113
UB6	199.879	162.111	249.114
UB7	50.278	38.87	61.622
UB8	200.758	167.112	245.113
UB9	300.317	228.119	370.609

Figure 11: Summary of Overall Response Time (For Figure 8.3).

The response times for each user base are shown in Figure 11, along with the total response times. Response time is used as the measurement in this specific algorithm iteration. Utilizing a network of data centers dramatically decreases response times overall and for each user base serviced when compared to using a single data center concept (Figures 9, 11).

IX.CONCLUSION

As a strategy to augment the accessibility of data within cloud data centers, an effective replica creation approach has been proposed. To ensure an optimum environment for customers, the data center's configuration is initially undertaken by an certified manipulator. Within the replica creation module, data replicas are produced based on their utilization frequency. The algorithm was formulated to streamline the deployment of replicas, consequently causing a deceleration in the reaction. Our upcoming endeavors drive focus happening refining an efficient method for replica management to reduce both costs and the requisite storage space.

REFERENCES

[1]. (2013) "Managing Data Replication and Placement Based on Availability" by Bakhta Meroufel and Ghalem Belalem 147–155 in AASRI Procedia, Volume 5.

[2]. "Federation in Cloud Data Management: Challenges and Opportunities," by Gang Chen, H.V. Jagadish, Dawei Jiang, David Maier, and Beng Chin Ooi (2014) IEEE Transactions on Knowledge and Data Engineering, Volume 26, Number 7, Pages 1670–1679.

[3]. "Optimal replica placement in hierarchical Data Grids with location assurance," J. Parallel Distrib. Comput. No.68, pp. 1517–1538; Jan-Jan Wu, Yi-Fang Lin, and Pangfeng Liu (2008)

[4]. "Survey of Techniques and Architectures for Designing Energy-Efficient Data Centers" IEEE Journal, Vol. 6, pp. 1231–1245; Junaid Shuja, Kashif Bilal, Sajjad A. Madani, Pavan Balaji, and Samee U. Khan (2015).

[5]. "Optimizing expansion techniques for ultra scale cloud computing data centres" Simulation Modeling Practice and Theory, Vol.12, pp.1569–1584, Mahmoud Al-Ayyoub, Mohammad Wardat, and Yaser Jararweh (2015)

[6]. Ishita Trivedi, Sanjeev Sharma, and Sanjay Bansal The International Journal of Distributed and Parallel Systems (IJDPS) published "A Dynamic Replica Placement Algorithm to Enhance Multiple Failures Capability in Distributed System" in its Vol. 2, No. 6 issue in November 2011.

[7]. Distributed Metadata Management Scheme in HDFS, Mrudula Varade and Vimla Jethani (2013) International Journal of Scientific and Research Publications, Vol. 3, Issue. 5, 1 ISSN, pp. 2250-3153.

[8]. "A dynamic replica management technique in data grid," Najme Mansouri and Gholam Hosein Dastghaibyfard (2012) 35th volume of Journal of Network and Computer Applications, pages 1297–1303.

[9]. Replica selection strategies in data grid, Reda Alhajj, Ken Barker, Rashedur M. Rahman (2008), J. Parallel Distributed Computing, No.68, pp. 1561–1574.

[10]. In their 2015 paper "An Energy Efficient QoS Based Replication Strategy," Reena S. More and Prof. Nilesh V. Alone Vol. 3, No. 6 of the International Journal of Innovative Research in Computer and Communication Engineering

[11]. Complete and fragmented replica selection and retrieval in Data Grids, Ruay-Shiung Chang and Po-Hung Chen, Future Generation Computer Systems No.23, pp. 536-546, 2007.

[12]. "MORM: A Multi-Objective Optimized Replication Management technique for Cloud Storage Cluster," Sai-Qin Long, Yue-Long Zhao, and Wei Chen (2014) pages. 234–244 of Journal of Systems Architecture, Volume 60.

[13]. "Fuzzy Authorization for Cloud Storage," IEEE Transactions on Cloud Computing, Vol. 2, No. 4, pp. 422-436, by Shasha Zhu and Guang Gong, Fellow, IEEE.

[14]. "Disaster-Aware Data centre Placement and Dynamic Content Management in Cloud Networks" IEEE Vol. 7, No. 7/J. Opt. Commun. Netw. 1943-0620, pp.681-695. [15] "A cloud approach to unified lifecycle data management in architecture, engineering, construction, and facilities management: Integrating BIMs and SNS." Advanced Engineering Inf.

[15]. Information Systems, Vol. 48, pp. 274–278. YoungJinChoi, SangHakLee, JinHwanKim, YongJuKim, HyeonGyuPak, GyuYoungMoond, JongHeiRa, and Yong-GyuJung (2015) "The approach to secure scalability and high density in cloud data-center."

AUTHORS:

Jeyaraman Sathiamoorthy is currently working as a Professor RMK ENGINEERING COLLEGE, Kavarapettai in Chennai. He has completed M. Tech (CIT) and Ph. D from Manonmaniam Sundaranar University, Tirunelveli. He has 20 years of teaching experience and she has published papers in Network Security in National and International journals and has presented in International Conferences and Seminars. His current area of interest is Programming Languages, Algorithms and ad-hoc networks especially MANET, VANET, FANET and Underwater Communication.

M. Usha is currently working as a Professor cum Director in MEASI Institute of Information Teahnology, Chennai. She has completed M.C.A. and M. Phil in Computer Science from Bharathidasan University, Trichy. She has also done her M. Tech (CIT) and Ph. D from Manonmaniam Sundaranar University, Tirunelveli. She has 20 years of teaching experience and she has published papers in Network Security in National and International journals and has presented in International Conferences and Seminars. Her current area of interest is Operating Systems, Algorithms and ad-hoc networks especially MANET, VANET, FANET and Underwater Communication.

P. Senthilraja is a research scholar in Anna university. He has completed M.Tech(CIT) from Manonmaniam Sundaranar University, Tirunelveli. He has 12 years of teaching experience and he has published papers in Network Security in National and International journals. His current area of interest is Machine learning, network security and cloud computing.