International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 4 (2017), pp. 583-594
© Research India Publications
http://www.ripublication.com

Enabling Geographically Distributed Cloud Applications using Key Value Storage Objects

P.Jyotheeswari [#1], J.Janet [#2], and T. Sunil Kumar Reddy [#3]

[#] *Department of CSE, Sri Venkateswara College of Engineering and Technology Chithoor, Andhra Pradesh, India.*

Abstract

In the recent years, the cloud storage gain popularity in the field of distributed services. Many applications need storage services to preserve the data securely. In this paper, the proposed model uses the key value based storage for cloud using the distributed hash table. The key value storage uses the operations such as store, retrieve and delete for the storage objects. The proposed model reduces the complexity of identifying the neighbouring nodes for replication. It has a special mechanism for searching the neighbouring nodes. The experimental evaluation is conducted with public key storage and the value size of the storage object. The results show the proposed key value storage has the superior performance when compared to the other mechanisms.

Keyword: Cloud Storage, Distributed Hash Table, Key value storage, Network.

I. INTRODUCTION

Cloud computing is an emerging filed in the recent years. Cloud computing provides different services to the clients in the rental basis. Cloud storage is one of the services provided by the cloud providers. Cloud storage virtualizes the resources to form the virtual resource group for easy accessing of data by the clients. This type of practice increases the resource utilization rate and provides the clients satisfaction. The major advantages of cloud storage are high resource utilization rate, low cost, rapid service, network access and good scalability. The clients need not bother about the resource constraints.

In the cloud computing system, the technology of virtualization spans the IT architecture, making all the system virtualized, including servers, storages, networks, applications, and so on. It realized the unified management, along with the monitor and control of all the resources, which increase the flexibility of the whole system and maximize its efficiency as well. At the same time, virtualization is a key technology to solve the issue of unified management of devices and to realize the flexibility of resource scheduling.

Decentralized attribute based encryption [1] does not rely on single central authority and can eliminate the distributed collaborative communication. Storage mechanism is crucial for cloud computing network; they should provide services to the large number of scattered data access nodes and data I/O in random access form. The authors [2] proposed similar group mining based on frequent sequence mining. In [3], authors proposed a new kind of hidden mapping hyper combined public key management scheme based on the hyper elliptic curve crypto systems; it solves the problem of large scale key management and storage issues in cloud storage. Data sharing [4] using crypto systems, auditing with key exposure resistant [5] and a special erasure coding schemes [6] for cloud storage systems described many optimization techniques for effective bandwidth management, sharing and high reliability in the data access and management.

This paper presents the key value based storage for applications in the geographically distributed cloud. The proposed model provides the flexibility for the applications to adjust their storage requirements based on their utility. For instance, an application can change their storage based on the past access history, time delay and number of replicas. Therefore, the proposed method has access control schemes, storage lifetimes, replication schemes, and access methods.

The rest of the paper is organized as follows. Section 2 describes about the related work regarding the cloud storage mechanisms. Section 3 explains about general architecture for cloud storage and sharing. Section 4 describes about the proposed model that deals with key value storage in cloud. Section 5 explains about the experimental evaluation of the proposed model with the existing model. Finally, conclusion of the paper is made in Section 6.

II. LITERATURE SURVEY

There is a vast literature on cloud storage that is similar to our research. To access the operations of UNIX file system, watch dogs [7] was introduced to change the file access permissions. Many researchers are made a contribution for integrating the CPU with the storage disks which provide the flexibility of on board applications. For instance, image processing, decision support systems and data mining [8, 9, and 10].

Distributed Hash Tables (DHTs) are extensively used in the field of distributed applications. For instance, distributed resource sharing, file sharing, distributed search engines, data centres and multi cast systems are the distributed applications uses the DHTs. Among the distributed applications, CFS[11], PAST[12] and i3[13] uses the store and retrieve operations, but other applications such as CoralCDN[14], Mercury[15], Bayeux[16] and Scribe[17] uses the user defined operations by modifying the underlying DHT values.

In the recent years, Abu et al. [18] proposed the robust cloud storage mechanism using key value based storage objects. This mechanism uses the RAID techniques for key value storage. The proposed mechanism so far called as RACS. The working procedure of the RACS is quite different from our mechanism. RACS uses the proxy for connecting the client to the key value storage. The major drawback of the RACS is usage of proxy. In [19], Bassani et al. proposed the DepSky for efficient storage purpose; the DepSky is a distributed storage system which has the special mechanism of cryptography and eraser coding to store the data. The DepSky permits only single write function in to storage.

To address the issue of space complexity and the durability of the storage, Chockler and Malkhi proposed the method for managing the number of clients and use the key value storage objects to carry the operations [20]. In [21], Ye et al. proposed the method for addressing the GC racing problem which serves the read operation. This method provides the writable access to the reader which is not required.

III. ARCHITECTURE FOR CLOUD STORAGE AND SHARING

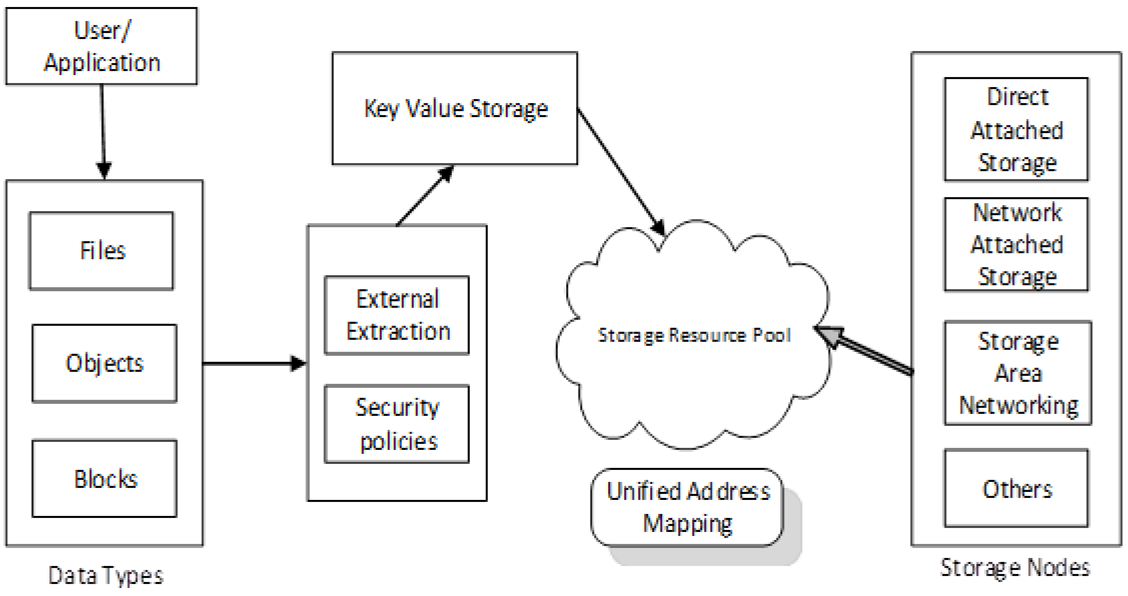

Fig.1. Proposed Cloud Storage architecture based on key value management

Fig.1. shows the high-level architecture of our cloud distributed storage system. The cloud storage system consists of three basic components. As the initial step the data is gathered in type of files, objects or blocks. These data is injected by the users or applications in to the cloud resources. The data type management is done by processing the security policies and external extraction of the data, which implements the value/node mapping, allowing a client to find nodes that store specific data items. The cloud storage system stores Storage Objects (SO). An objects consists of a key and its associated state (i.e., a value, stored as an untyped byte string), along with optional code that operates on that state. The code is structured as a set of handlers that specify how the object behaves, i.e., how it modifies its state when certain events occur.

Clouds storage resource pool is grouped by network, file systems and various storage components. Since clouds are presented as a service, they are referred to as a storage service system. Cloud storage architectures may be comprised of network devices, storage pools, file systems, service-level interfaces and common access interfaces, etc. It is made up of distributed resources, but acts as a single resource, hence providing high fault tolerance through data redundancy and distribution.

The infrastructure of a cloud storage resource pool mainly consists of storage components such as storage devices, servers, and storage networks (NAS, SAN, FC, iSCSI, SCSI, SAS, etc.). It also comprises of wired/wireless networks.

In the storage management layer, there can be a storage manager or a management system which administers or advocates the management of storage in the infrastructure layer. Its most important role is data management. This data management collaborates between multiple domains to ensure data redundancy, fault tolerance, load balance and storage maintenance.

IV. PROPOSED METHOD FOR KEY VALUE STORAGE

The proposed distributed key value storage is defined with two elements; one is key element and other one is value which is associated with the key. The pair can be simply represented as {key, value}. Fig.2 shows the functionality of the key value storage in distributed cloud.

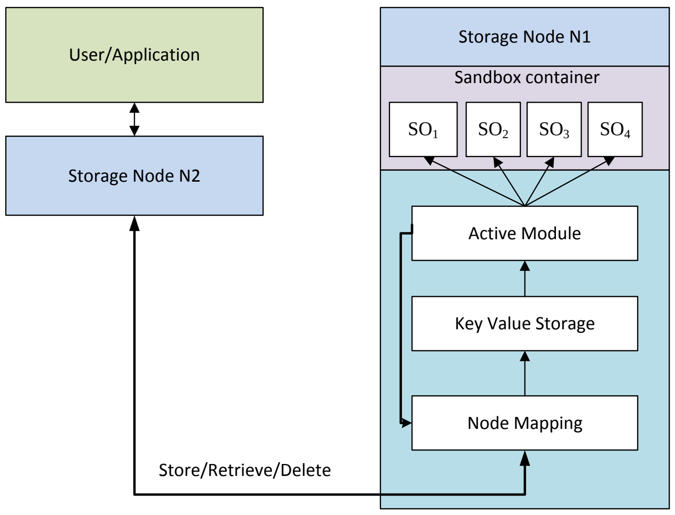

Fig.2. Functionality of proposed method

As shown in Fig.2, the proposed method has three modules; node mapping, key value storage ad active module.

Node Mapping: In the node mapping module, the clients has the accessibility to find the nodes for storing the data items. If the nodes following the distributed hash table (DHT), the node mapping module computes the hash function for the keys to retrieve the IDs of the concerned nodes with their value.

Key value Storage: In the second module, the key value storage is used to the store the key pairs {key, value} of each node. This module usually exports the store/retrieve interface. In general, the public key value storage stores the untyped byte strings but, in the proposed method, it stores the storage objects (SO). The storage object contains the key and its associated value, the value contains the untyped byte string.

Active Module: The active module is associated with the storage object invocations, execution environment and security policies. As an initial stage, the user application which is running on the remote node invokes the storage objects using the storage method. When the store or retrieve method is invoked by the node, then automatically checksum is calculated for the code which is associated with the SO.

A. *Storage Object (SO)*

The storage objects support the invocations of methods such as store and retrieve as well as onTimer. The onTimer method is used for custom replication with in the storage object. In Fig.3, it is shown that how the invocations can be handled with in the proposed method.

```
Retrieve_SystemTime() /* for node's access time*/

Retrieve_IP()  /* for node's IP*/

Retrieve_ID()  /* for node's Distributed Hash Table ID */

Retrieve_Key() /* for Storage Object Key*/

Retrive(Key, {values}) /* for retrieving the replicas of the values stored in different nodes*/

Delete_self() /* deletes the Storage Object*/

Store(Key, values, [nodes]) /* stores the keys and values of the nodes*/

Search(key) /* searches for the nearest nodes to the key*/
```

Fig. 3. Storage objects API

The storage object APIs are used to handle the communication outside of the sandbox. The proposed method supports two types of operations, one is to gather the information of the local node and another one is to execute the storage operations.

B. Algorithm for Key Value Storage

Key value storage in the cloud maintains the array for storing the pair of keys and values. The array is functioned with storage, retrieves and deletes the keys in a set of X and associated set of keys K. In the proposed algorithm, the key value storage supports three operations such as storing a value with associated key (store {Key, α}), Retrieving a value associated with key (α ->retrieve {key}) and delete the value with the associated key (delete {key}).

The size of the value is much bigger than the key in the pair, so the values in the set X cannot be interchanged as the key in the set K. In algorithm 1, the operations of key value storage is mentioned with the object ‘i’ and Y is the variable which is associated with the keys and values.

Algorithm 1: Cloud based key value storage

```
Begin
Initialize X, K -> {$\phi$}
Y $\subseteq$ {K,X}
On initialize store$_i${Key, Value}
    Y <-(Y\ {(Key, α)| α $\in$X}) $\cup$ (key, value)
    Return ACK
On initialize Retrive$_i${key}
    If $\exists$α: (Key, α) $\in$Y then
        Return α
    Else
        Return ‘α doesn’t exist’
On initialize Delete$_i${key}
    Y <-(Y\ {(Key, α)| α $\in$X})
    Return ACK
End
```

C. Monitoring the Storage Object

The storage objects maintain the functionalities of the read or update of the objects. These types of practices are helpful in reducing the errors in the storage. In Fig.4, Pseudo code explains about the storing and accessing the values in the nodes.

```
hostIps, replicaIPs. accessorIps={ϕ}
On Retrieve(CallerIp)
  DHT.insert(self.accessorIps, CallerIp)
  Return self
On Store(caller)
  DHT.insert(self.accessorIps, Caller. RerieveIp)
  DHT.insert(self.hostIPs,DHT.node.RetrieveIP)
  Return self
On Timer()
  DHT.Store(DHT.Retrievekey(), self, 40)
```

Fig.4. Pseudo code for monitoring the SO

D. Measurement of DHTs

Storage objects (SOs) provides the opportunities to measure the DHTs with the help of registered DHT nodes. The measurement of DHTs provides complete view of DHT properties like node lifetime, churn, inconsistencies etc.
For example, to track the closest nodes to the storage object, the pseudo code in Fig. 5 is used.

```
so.k={}
function so:handlelookup(n)
        self.k [DHT.RetrieveSysTime()]=n
end
function so:onTimer()
        DHT.lookup(DHT.RetrieveKey(), self.k)
end
```

Fig.5. pseudo code to measure the Lifetime of the node

In Fig.5, ‘n’ is the number of nodes, k is the neighbouring nodes and DHT is the distributed hash table.

V. EXPERIMENTAL EVALUATION

In order to test the efficiency of the proposed model, we simulate the proposed model and key management scheme based on the public key infrastructure. Fig. 6 shows the situation of network delay in terms of number of nodes. It can be observed that, the scheme with public key infrastructure, network delay is increased drastically due to the complexity of the public key infrastructure. The proposed scheme works superiorly when compared to the public key infrastructure. The simulation is carried with the history records; the proposed key based updating algorithm is compared with the classical key updating algorithm. The proposed algorithm shows 24% of reduction in time when compare to the classical key updating algorithm. Fig. 7 shows the time delay of key update as the node number changes in both the algorithms.

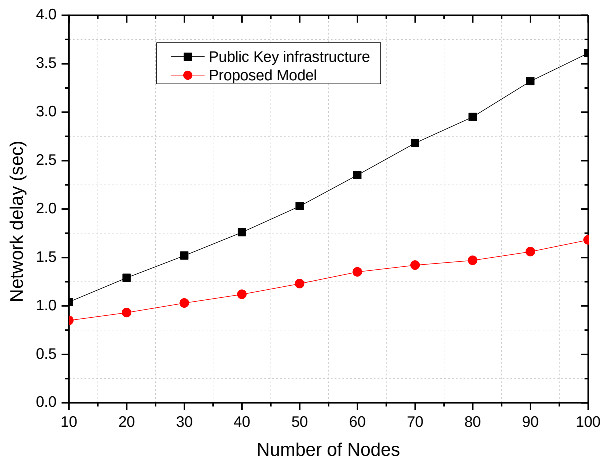

Fig.6. Comparison of network Time delay in terms of nodes using proposed model

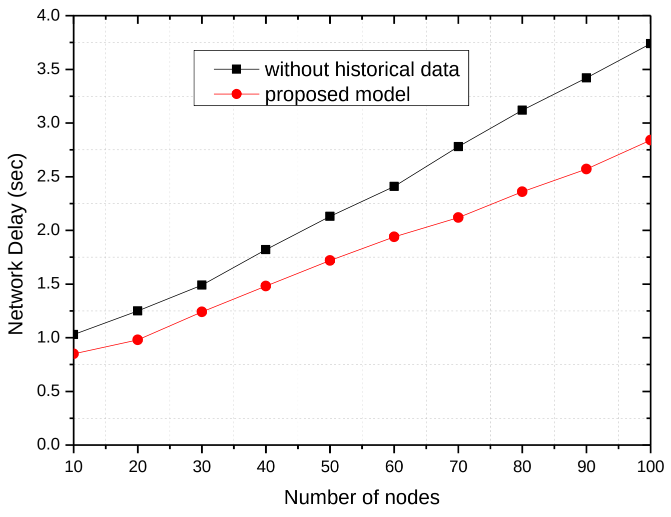

Fig.7. The time delay for key update as the node number changes

The performance of the read and write operations in the key value storage of proposed method is shown in Figs. 8 and 9. By observing the Fig. 8, it is clear that the write throughput of proposed method is high when compared to without proposed method and in Fig. 9, the read throughput is better for the proposed method.

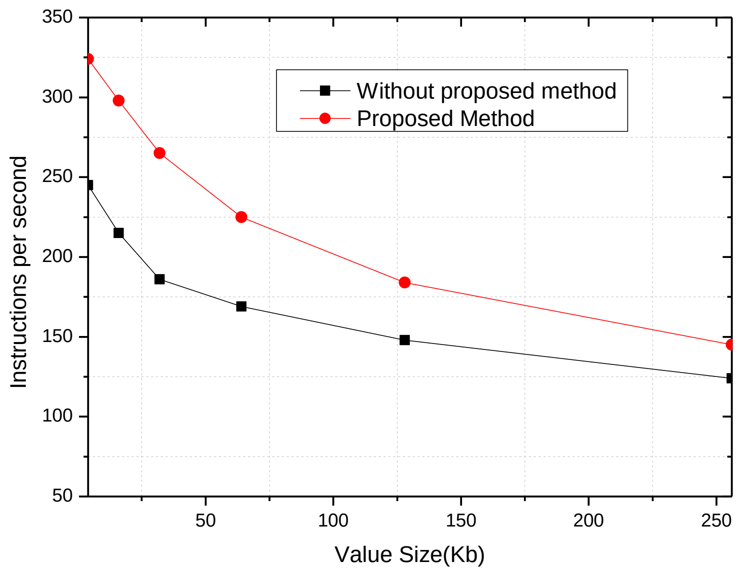

Fig.8. Write Throughput for proposed method

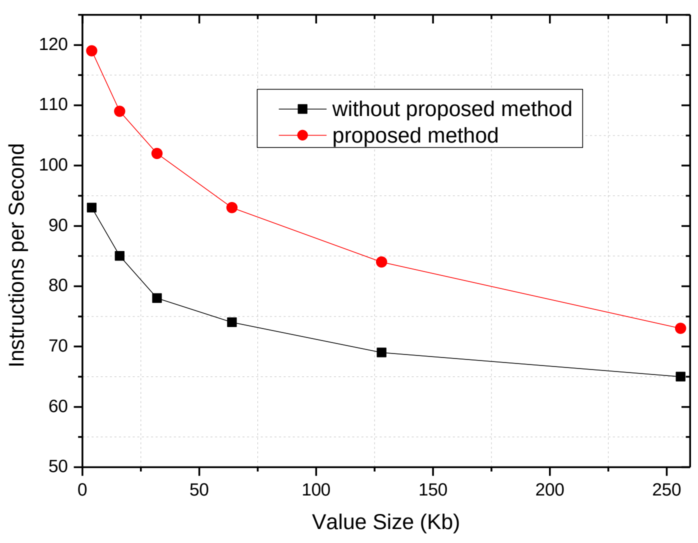

Fig.9. Read Throughput for proposed method

VI. CONCLUSION

In this paper, we presented the novel mechanism for cloud storage using the key value storage object. The proposed method utilized the set of keys and values for storing the node data. The operations of the key value storage are developed for store, retrieve or delete the values of the nodes in the distributed hash table. The proposed model reduces the complexity of identifying the neighbouring nodes for replication. It has a special mechanism for searching the neighbouring nodes. The proposed key value storage is tested with different conditions. It is proved that, the proposed method has superior performance in terms of read and write operations as well as the node lifetime also increased.

ACKNOWLEDGMENT

This work was supported in part by Government of India under women scientist scheme A (WOS-A) grant number SR/WOS-A/ET-86/2012(G)

REFERENCES

[1] Jianwei Chen; Huadong Ma, "Efficient decentralized attribute-based access control for cloud storage with user revocation," Communications (ICC), 2014 IEEE International Conference on , vol., no., pp.3782,3787, 10-14 June 2014.

[2] Hongtao Du; Zhanhuai Li, "Data rearrange based on mining block access sequence in Cloud Storage," Computer Science and Network Technology (ICCSNT), 2011 International Conference on, vol.4, no., pp.2507,2511, 24-26 Dec. 2011.

[3] Ningning Song; Yueyun Chen, "Novel hyper-combined public key based cloud storage key management scheme," Communications, China , vol.11, no.14, pp.185,194, Supplement 2014.

[4] Cheng-Kang Chu; Chow, S.S.M.; Wen-GueyTzeng; Jianying Zhou; Deng, R.H., "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage," Parallel and Distributed Systems, IEEE Transactions on , vol.25, no.2, pp.468,477, Feb. 2014.

[5] Jia Yu; KuiRen; Cong Wang; Varadharajan, V., "Enabling Cloud Storage Auditing With Key-Exposure Resistance," Information Forensics and Security, IEEE Transactions on , vol.10, no.6, pp.1167,1179, June 2015.

[6] Li, J.; Li, B., "Erasure coding for cloud storage systems: A survey," Tsinghua Science and Technology , vol.18, no.3, pp.259,272, June 2013.

[7] B. N. Bershad and C. B. Pinkerton. Watchdogs – extending the UNIX file system. Computer Systems, 1(2), 1988.

[8] K. Keetong, D. Patterson, and J. Hellerstein. A case for intelligent disks (IDISKs). ACM SIGMOD Record, 27(3), August 1998.

[9] E. Riedel, G. Gibson, and C. Faloutsos. Active storage for large scale data mining and multimedia. In Proc. of 24th International Conference on Very Large Databases, August 1998.

[10] A. Acharya, M. Uysal, and J. Saltz. Active disks: Programming model, algorithms and evalaution. In Proc. of the 8th Conference on Architectural Support for Programming Languages and Operating Systems, October 1998.

[11] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In Proc. of SOSP, 2001.

[12] A. Rowstron and P. Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In Proc. of SOSP, 2001.

[13] I. Stoica, D. Adkins, S. Zhuang, S. S. nker, and S. Surana. Internet indirection infrastructure. In Proc. of SIGCOMM, 2002.

[14] M. J. Freedman, E. Freudenthal, and D. Mazi`eres. Democratizing content publication with coral. In NSDI, pages 239–252, 2004.

[15] A. R. Bharambe, M. Agrawal, and S. Seshan. Mercury: Supporting scalable multi-attribute range queries. In Proc. of SIGCOMM, 2004.

[16] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz. Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination. In Proc. of NOSSDAV, 2001.

[17] A. I. T. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel. Scribe: The design of a large-scale event notification infrastructure. In Proc. of the Third International COST264 Workshop on Networked Group Communication, 2001.

[18] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon, “RACS: a case for cloud storage diversity,” in SoCC’10, 2010.

[19] A. Bessani, M. Correia, B. Quaresma, F. Andr´e, and P. Sousa, “Depsky: Dependable and secure storage in a cloud-of clouds,” in EuroSys’11, 2011.

[20] G. Chockler and D. Malkhi, “Active disk Paxos with infinitely many processes,” Dist. Comp., vol. 18, pp. 73–84, 2005.

[21] Y. Ye, L. Xiao, I.-L. Yen, and F. Bastani, “Secure, dependable, and high performance cloud storage,” in SRDS’10, 2010.

AUTHOR PROFILE

	P.Jyotheeswai received the B.Tech degree in computer science & Engineering from Jawaharlal Nehru & Technology University, Hyderabad, in 2005 and the M.Tech degree in Dr M.G.R University, Chennai and pursuing Ph.D. degree in computer science from Vellore Institute of Technology, Vellore. She is currently an Associate Professor in the Department of CSE in SVCET, Chittoor . Her research interests include security ,cloud computing, data mining, Distributed systems, Big data Analytics
	Dr T. Sunil Kumar Reddy is a professor in the Department of Computer science & engineering, Sir Venkateswwara college of Engineering and Technology, Chittoor. He received the B.Tech degree in Information Technology from Satyabhama University in 2005, the M.Tech degree in Information Technology from V.I.T University in 2007and PhD degree in computer science & engineering from JNTUA University, Anantapur. His research interests include Cloud Computing, High performance computers, Wireless Networks.
	Dr. Janet. J received her B.E. and M.E Degree in Computer Science & Engineering from Madras University. She has completed his PhD from Sathyabama University, Chennai. Now she is working as a Professor in the Department of Computer Science & Engineering in Sri Venkateswara College Of Engineering & Technology, Chitoor, Andhra Pradesh. Her research interests include Image Processing, Data Mining, and Networks, cloud computing.