Efficient Cross User Client Side Data Deduplication in Hadoop

Priteshkumar Prajapati[1]*, Parth Shah[1], Amit Ganatra[2], Sandipkumar Patel[1]
[1] Department of Information Technology, C.S.P.I.T., CHARUSAT, Anand, India.
[2] Department of Computer Engineering, C.S.P.I.T., CHARUSAT, Anand, India.

* Corresponding author; email: pritesh.pnp.007@gmail.com
Manuscript submitted January 11, 2016; accepted April 14, 2016.
doi: 10.17706/jcp.12.4.362-370

Abstract: Hadoop is widely used for applications like Aadhaar card, Healthcare, Media, Ad Platform, Fraud Detection & Crime, and Education etc. However, it does not provide efficient and optimized data storage solution. One interesting thing we found that when user uploads the same file twice with same file name it doesn't allow saving the same file. But when user uploads the same file content with different file name Hadoop allows uploading that file. In general same files are uploaded by many users (cross user) with different name with same contents so this leads to wastage of storage space. So we provided the solution of above problem and provide Data Deduplication in Hadoop. Before uploading data to HDFS we calculate Hash Value of File and stored that Hash Value in Database for later use. Now same or other user wants to upload the same content file but with same content, our DeDup module will calculate Hash value and verify it to HBase. Now if Hash Value is matched so it will give message that "File is already exits". Experimental analysis demonstrates (i.e. Text, Audio, Video, Zip files etc.) that proposed solution gives more optimized storage acquiring very small computation overhead and having optimized storage space.

Key words: Cloud storage, deduplication, Hadoop, Hadoop distributed file system, Hadoop database.

1. Introduction

Cloud computing is the most in demand advanced technology being utilized throughout the world. It is one of the most significant research ideas whose application is being researched recently. One of the prominent services offered in cloud computing is the cloud storage. With the cloud storage, data is stored on multiple third party servers, rather than on the dedicated server used in traditional networked data storage. All data stored in multiple third party servers are not concern by the user and no one knows where exactly data saved [1]. CloudMe [2], CrashPlan [3], Dropbox [4], Mozy [5], Team Drive [6], UbuntuOne [7] and Wuala [8] offer services for information storage, information access and different process capabilities in a reliable manner.

Cloud storage is a model of networked enterprise storage where data is stored not only in the user's computer, but also in virtualized pools of storage which are generally hosted by third parties. Hosting companies operate large data centers, and end user who require their data to be hosted, buy or lease storage capacity from them. The center operators, in the background, virtualize the resources according to the requirements of the customer and expose them as storage pools, which the customers can themselves use to store files or data objects and physically, the resource may span across multiple servers and the

safety of the files depends upon the hosting websites [9].

A distributed system consists of a collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility [10]. Distributed data storage is a computer network application where information is stored on more than one node, often in a replicated fashion. It is usually referred as either a distributed database where users store information in a number of nodes, or a computer network application in which users store information in a number of peer network nodes [11].

Hadoop is high-performance distributed data storage and processing system [12]. Two major subsystems of Hadoop are HDFS (for storage), and Map-Reduce (for parallel data processing). The Data Deduplication technology is widely used in Business File Server, Database, Backup Devices or lots more storage devices

Data Deduplication is the process of identifying the redundancy in data and removing it. It is found that Deduplication technique can save up to 90% storage, dependent on applications [13]. Deduplication has proven a highly effective technology in eliminating redundancy in backup data.

Hadoop doesn't provide effective Data Deduplication solution. Assuming a popular video or movie file is uploaded to HDFS by one million users and stored into three million files through Hadoop replication and thus it is wasting of disk space. Through proposed system, only single file spaces are occupied, namely reaching the utility of completely removing duplicate files.

The rest of the paper is organized as: Section 2 gives the outlines the related work available in literature; Section 3 describes the proposed system; Section 4 shows the implementation details and analysis of the solution; conclusion and future work is followed in Section 5.

2. Literature Review

Hadoop [12], [14] is an open source software application that consists of two main components: HDFS (Hadoop Distributed File System) for Storage and MapReduce for distributed computing.

While storing a file, HDFS break files into smaller blocks (like 64 MB, for instance) and store the individual blocks in multiple servers. Hadoop ensures that each block is stored in at least three nodes. Though this process increases the total space required to store data, it gives a good amount of redundancy as data can be recovered and reconstructed. MapReduce function of Hadoop takes advantage of this distributed storage functionality to provide distributed computing capability. Large file is split into various smaller blocks and distributed across individual servers. Multiple blocks are processed simultaneously using the computing power of individual servers and all their output is integrated by a master server to create the final output which can be presented to the user [14].

HBase [15], [16] is a column family-oriented database: It's often described as a sparse, distributed, persistent, multidimensional sorted map, which is indexed by rowkey, column key, and timestamp. HBase stores structured and semi structured data naturally so you can load it with tweets and parsed log files and a catalog of all your products right along with their Customer reviews. It can store unstructured data too, as long as it's not too large. It doesn't care about types and allows for a dynamic and flexible data model that doesn't constrain the kind of data you store. HBase isn't a relational database. It is not like SQL or enforces relationships within your data. It doesn't allow inter row transactions, and it doesn't mind storing an integer in one row and a string in another for the same column. HBase is designed to run on a cluster of computers instead of a single computer. The cluster can be built using commodity hardware; HBase scales horizontally as you add other machines to the cluster. Each node in the cluster provides a bit of storage, a bit of cache, and a bit of computation as well. This makes HBase incredibly flexible. No node is unique, so if one of those machines breaks down, you simply replace it with another.

There are some issues in Hadoop like Security, Network Level Encryption and Deduplication which are briefly explained below:

Security: Security is also one of the major concerns because Hadoop does offer a security model but by default it is disabled because of its high computation complexity [17].

Network Level Encryption: Hadoop does not offer storage or network level encryption which is very big concern for government sector application data [17].

Deduplication: Deduplication is able to reduce the required storage capacity since only the unique data is stored [18]. Hadoop does not provide Deduplication solution.

In Reference [19] shows that, it provides the framework in which service level agreement has been used as the common standard between user and provider to ensure data security in the cloud storage system. This technology is divided into three parts: storage protect, transfer protect and authorize. The main advantage is administrator can view encrypted pieces of files saved in cloud storage.

In Reference [20] shows that, it provides synchronous and asynchronous backup using Deduplication with file level, block level and byte level to save users data storage space and cut the cost of storage. As the system uses variable-length data blocks Deduplication which is the most efficient data de-duplication technology considered now a days.

In Reference [21] shows that, proposed algorithm for an efficient indexing mechanism using the advantage of B+ tree properties. In this algorithm, File is divided into variable-length chunks using Two Thresholds Two Divisors chunking algorithm. Chunk IDs are calculated by applying SHA hash function to the chunks. Chunk IDs are used to build as indexing keys in B+ tree like index structure is used to avoid full-chunk indexing to identify the incoming data is new, which is time consuming process. The main advantage is, Searching time for the duplicate file chunks reduces from $O(n)$ to $O(\log n)$.

Some other factors like reduce backup data storage, reducing storage capacity, space and energy consumption solution provided in [22]. Implementation is divided into foreground and background processing. Foreground processing is done through means of pure software implementation and background processing method used to achieve integrated software and hardware equipment. The main advantage is, File level Deduplication has generally less than 5:1 compression ratio. Block-level storage technology can compress the data capacity of 20: 1 or even 50: 1.

In Reference [23] shows that, they developed backup framework PRUNE, which effectively achieves the inter file and intra file information redundancy. Filter based main memory index lookup structure have been used to effectively eliminate redundancy and perform efficient backup. The main advantage is 99.4% of disk access involved in fingerprint management has been eliminated by PRUNE.

In Reference [24] shows that, they proposed KEYGEN, UPLOAD, AUDITINT, DEDUP to provide Proof of storage with Deduplication fulfill data integrity and duplication at the same time. KEYGEN is the key generation algorithm. UPLOAD is the data uploading protocol running by a client and a server over a private channel so that privacy of the data is assured. AUDITINT is the data integrity auditing protocol. It is executed between server and auditor so that server convinces auditor that integrity of some data file stored in the cloud is assured. DEDUP is the Deduplication checking protocol. It is executed between server and client, who claim to own a data file. The main advantage Proof of Storage with Deduplication scheme incurs smaller communication overhead. But it's having the disadvantage that Proof of Storage with Deduplication scheme is slightly less efficient than the Proof of Ownership scheme.

In Reference [25] shows that, they proposed KEYGEN, MODIFIED UPLOAD, AUDITINT, and DEDUP to provide insecurity of the Proof of Storage with Deduplication scheme under new attack model that malicious client's activity is to dishonestly use the keys. KEYGEN is the key generation algorithm. MODIFIED UPLOAD is the data uploading protocol running by a client and a server over a private channel so that

privacy of the data is assured. AUDITINT is the data integrity auditing protocol. It is executed between server and auditor so that server convinces auditor that integrity of some data file stored in the cloud is assured. DEDUP is the Deduplication checking protocol. It is executed between server and client, who claim to own a data file. The main advantage is an improved scheme of combination client's keys with the server contributed random values to mitigate the attack. But it's having the disadvantage that strength of the Proof of Storage with Deduplication scheme stands on the strong assumption that all clients are honest in terms of generating their keys.

In Reference [26] shows that, they provide enhanced and generalized convergent encryption method used for security concern in cross-user client-side Deduplication of encrypted files with bounded leakage model in cloud storage. The main advantage is Cross-user client-side Deduplication scheme with data confidentiality in more secure way. But it's having the disadvantage that before scheme starts to execute, it is allowed for one-time bounded leakage of a target file. Implementation is not optimized.

In Reference [27] shows that, they provide DupLESS (Duplicate Less Encryption for Simple Storage) in which provides a more secure, easily-deployed solution for encryption that supports Deduplication. The hash function SHA256 and the symmetric cipher AES128 encryption algorithms were used in DupLESS to provide secure outsourced storage that both supports Deduplication and resists Brute-force attacks. The main advantage is DupLESS provides security that is usually significantly better than current convergent encryption based Deduplicated encryption. Main overhead of DupLESS with respect to convergent encryption, has been optimized for low latency. But it's having the disadvantage that given the small constant size of the extra file sent by DupLESS, but overhead quickly reduces as files get larger. In [28]-[30] they provides more efficient solution in terms of time compare to DupLESS.

In Reference [31] shows that, they provide Dedoop tool for MapReduce-based entity resolution (ER) of large datasets. Dedoop supports a browser-based specification of complex ER workflows including blocking and matching steps as well as the optional use of machine learning for the automatic generation of match classifiers. Specified workflows are automatically translated into MapReduce jobs for parallel execution on different Hadoop clusters. To achieve high performance Dedoop supports several advanced load balancing strategies.

In Reference [32] shows that, they proposed a scalable parallel Deduplication algorithm called FERAPARDA. By using probabilistic record linkage, they were able to successfully detect replicas in synthetic datasets with more than 1 million records in about 7 minutes using a 20-computer cluster, achieving an almost linear speedup. They believe that their results do not have similar in the literature when it comes to the size of the data set and the processing time.

In Reference [33] shows that, they propose a scalable design of a distributed de-duplication system which leverages clusters of commodity nodes to scale-out suitable tasks of a typical de-duplication system. They explain their distributed duplicate detection workflow implemented in Hadoop's map-reduce programming abstraction. With the aid of proposed design, they can prevent the critical storage controller resources from being spent in a workload which can be easily scaled-out. The other important aspect of this design is that it allows for leveraging cheaper resources which are not at the controller but still are part of the storage tier (by using commodity hardware). This has an effect of lowering the overall cost of the solution, while allowing for higher I/O performance to be served through saved resources at the controller. Another interesting aspect of this design is that it involves non-shared coarse-grained parallelism which enables better scalability. Thus with this design, the number of concurrent de-duplication processes could be increased by an order of magnitude without taking additional resources at the controller, by adding more commodity hardware to the storage cluster.

3. Proposed System

Recently many Deduplication solutions have been proposed. The concept of cloud storage is derived from cloud computing. It refers to a storage device accessed over the Internet via Web service application program interfaces (API). HDFS (namely Hadoop Distributed File System, is a distributed file system that runs on commodity hardware; it was developed by Apache for managing massive data. The advantage of HDFS is that it can be used in a high throughput and large dataset environment.

As we are aware Hadoop has two main components: HDFS and MapReduce. HDFS Client provides interface between user and Hadoop. So when user wants to upload data in Hadoop following steps are performed:

1. HDFS Client communicates with NameNode (via heartbeat messages)
2. NameNode finds appropriate DataNode.
3. NameNode provides details of DataNode.
4. HDFS Client upload file to DataNode.
5. DataNode divides files into blocks and stores it. It makes by default Three Replicas of that file.
6. DataNode provides blocks details to NameNode.

So when user wants to download that file, HDFS Client communicates to NameNode and NameNode provides details of DataNode to HDFS Client. DataNode merge the blocks and it provide file for HDFS Client.

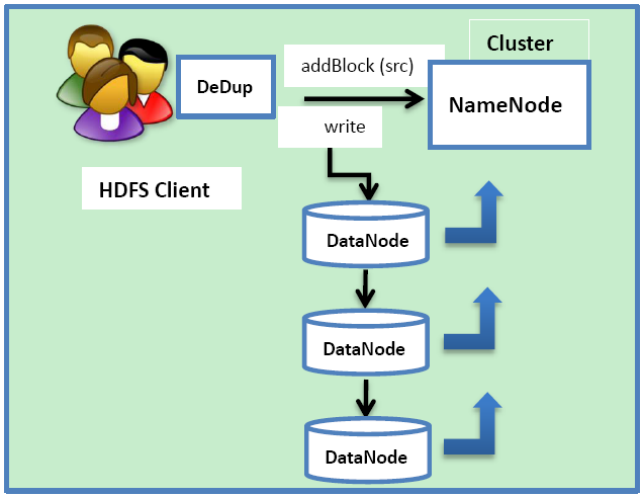

Fig. 1. Client side data deduplication in Hadoop (proposed system).

In Reference [29], [31] shows that, Hadoop gives message that file already exists when user upload the same file another time. (With same file name and same file content). But one interesting thing we found that when user uploads the same file content with different File Name Hadoop allows uploading that file. In general same files are uploaded by many users (Cross user) with different name with same contents so this leads to wastage of storage space. So we provided the solution of above problem. So we made changes in HDFS Client to provide Data Deduplication in Hadoop. Before uploading data to HDFS we calculate Hash Value of File and stored that Hash Value in Database for later use. As shown in Fig. 1 in the proposed approach, the main issue to be addressed is how to identify duplications and how to prevent duplicates from uploading to HDFS. For this issue, we use SHA algorithm to make a unique fingerprint for each file and set up a fast fingerprint index in HBase to identify the duplications. HBase is Hadoop database, which is an open-source, distributed, versioned, column-oriented database. It is good at real time queries. HDFS has been used in numerous large scale engineering applications. Based on these features, HDFS as a storage system and HBase as an indexing system are used in our work. So we made Deduplication module in HDFS client. When data will be uploaded for first time to HDFS, the Deduplication module will calculate hash

value of the file and store it in HBase and store the file in HDFS. When new data will be uploaded by any users, the system will calculate its hash value and will check in HBase that if the hash value already exists or not. If hash value exists, then the system will give the message that file/content already exist in HDFS and will not allow uploading file for second time and will not store any entry of that file in HBase. If hash value does not exist then the system calculates Hash value of new file and then put the entry of new file and its Hash value is stored in HBase and file is uploaded to HDFS.

4. Implementation of Proposed System

Experimental setup and results are performed on a computer having following features:

1) Intel(R) Core (TM) i5-2400 CPU @3.10 GHz processor 8 GB RAM, and Oracle Virtual Box version 4.3 in Ubuntu 12.04 LTS 32-bit using Hadoop 1.0.3, HBase-0.94.1, JDK_1.7.0_10, Eclipse_Juno in Windows 7 Professional - 64bit.
2) Intel(R) Core (TM) i5-2400 CPU @3.10 GHz processor 4 GB RAM, and Oracle Virtual Box version 4.3 in Ubuntu 12.04 LTS 32-bit using Hadoop 1.0.3, HBase-0.94.1, JDK_1.7.0_10, Eclipse_Juno in Windows 7 Professional - 64bit.

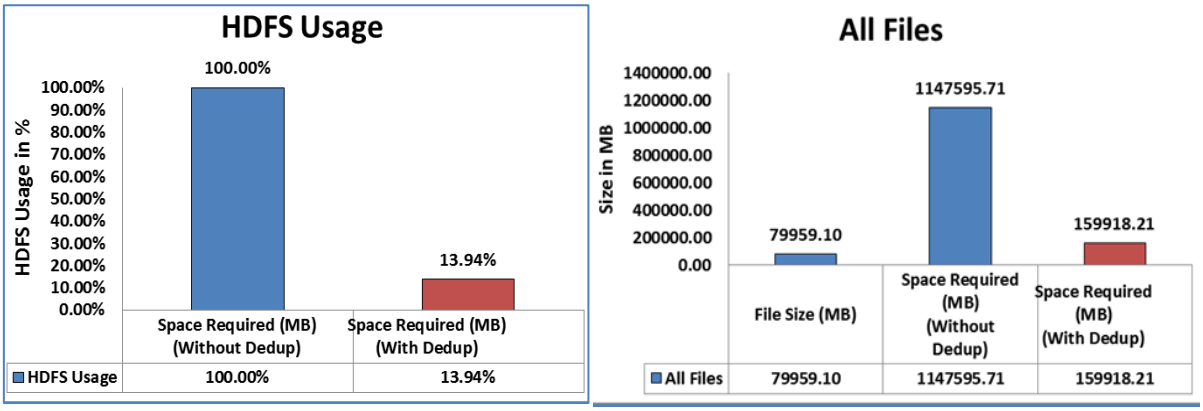

Fig. 2. Space requirement without deduplication and with deduplication.

In Fig. 2, for experimental results are shown using Virtual Box 4 GB RAM allocated for Master, 2 GB for all the slave machines. We have used Text (6.3 GB), Audio (1.28 GB), Image (1.17 GB), Video (63.49 GB) and Mix file (Zip) (5.75) kind of File Types for our Experiments. We created 1 Master Node and 50 slave machines to perform this Deduplication in our university with Replication Factor = 3. As per above testing 78.09 GB File size we wanted to store, So without our solution Hadoop requires around 1120.70 GB space but with our solution it required only 156.17 GB of data. So with our DeDup solution in Hadoop for above experiment it utilizes around 14 % HDFS storage space and saves nearly 86 % storage space.

5. Conclusion

The proposed system is designed to prevent duplicity of storage space in HDFS and to provide effective data storage solution. It provides cross user file level Deduplication at client side. This solution ensures that proposed system reduces bandwidth usage, since the duplicate files that already exist on storage server are detected and would not send it to the server. With the proposed system there is quite less communication overhead and overall performance of the system is significantly improved.

Our future work lies on use of proposed system to provide block-level Deduplication with confidentiality for Hadoop and to integrate it with efficient data recovery with data integrity protection [34].

Acknowledgment

This work was done in Chandubhai S. Patel Institute of Technology in CHARUSAT University at Changa. All this would not have been possible without the active support from Prof. Parth Shah, the Head of Department of Information Technology, whose encouragement, constant supervision and their guidance from the preliminary to the concluding level enabled me to develop an understanding of my work. He has always been willingly present whenever I needed the slightest support from him, I would like to thank all the faculties of the Department of Information Technology, CHARUSAT for their support. I would like to thank my parents and friends for their consistent emotional support and otherwise for being my continuing source of inspiration. Last but not least I would like to acknowledge all of my supportive and encouraging colleagues who made a significant contribution during each phase of project directly or indirectly.

References

[1] Kumar, A., Lee, B. G., Lee, H., & Kumari, A. (2012). Secure storage and access of data in cloud computing. *Proceedings of International Conference on ICT Convergence* (pp. 336-339).
[2] CloudMe. From www.cloudme.com
[3] CrashPlan. From www.crashplan.com
[4] DropBox. From www.dropbox.com
[5] Mozy. From www.mozy.com
[6] TeamDrive. From www.teamdrive.com
[7] Ubuntu One. From www.one.ubuntu.com
[8] Wuala. From www.wuala.com
[9] Cloud Storage. From http://www.definitions.net/definition/Cloud%20storage
[10] Distributed System. From http://encyclopedia2.thefreedictionary.com/Distributed+systems
[11] Distributed Data Store. From http://dbpedia.org/page/Distributed_data_store
[12] Hadoop. From http://hadoop.apache.org
[13] SNIA. *Understanding Data De-duplication Ratios*. White paper.
[14] Rajesh, K. (2011). Hadoop — Create distributed computing and scalable storage with entry level servers.
[15] Hbase. From http://www.hadoopuniversity.in/apache-hbase-training/
[16] HBase. From https://hbase.apache.org
[17] Hadoop Issues. From http://www.guruzon.com/6/introduction/hadoop/pros-and-cons-of-hadoop
[18] Data Deduplication. From http://netapp-blog.blogspot.in/2009_05_01_archive.html
[19] Zhang, X., Du, H. T., Chen, J. Q., Lin, Y., & Zeng, L. J. (2011). Ensure data security in cloud storage. *Proceedings of International Conference on Network Computing and Information Security* (pp. 284-287).
[20] Sun, G. Z., Dong, Y., Chen, D. W., & Wei, J. (2010, October). Data backup and recovery based on data de-duplication. *Proceedings of International Conference on Artificial Intelligence and Computational Intelligence* (pp. 379-382).
[21] Thwel, T. T., & Thein, N. L. (2009). An efficient indexing mechanism for data Deduplication. *Proceedings of International Conference on Current Trends in Information Technology* (pp. 1-5).
[22] He, Q., Li, Z., & Zhang, X. (2010). Data deduplication techniques. *Proceedings of International Conference on Future Information Technology and Management Engineering* (pp. 430-433).
[23] Won, Y., Ban, J., Min, J., Hur, J., Oh, S., & Lee, J. (2008). Efficient index lookup for De-duplication backup system. *Proceedings of IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems* (pp. 1-3).
[24] Zheng, Q., & Xu, S. (2012). Secure and efficient proof of storage with deduplication. *Proceedings of the Second ACM Conference on Data and Application Security and Privacy* (pp. 1-12).

[25] Shin, Y. J., Hur, J., & Kim, K. (2012). Security weakness in the proof of storage with deduplication. *IACR Cryptology ePrint Archive*, 1-11.

[26] Xu, J., Chang, E. C., & Zhou, J. (2013). Weak leakage-resilient client-side deduplication of encrypted data in cloud storage. *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security* (pp. 195-206).

[27] Bellare, M., Keelveedhi, S., & Ristenpart, T. (2013). DupLESS: Server-aided encryption for deduplicated storage. *Proceedings of the 22nd USENIX Security Symposium* (pp. 179-194).

[28] Prajapati, P., & Shah, P. (2014). Efficient cross user data deduplication in remote data storage. Proceedings *of International Conference on Convergence of Technology.*

[29] Prajapati, P., & Shah, P. (2015). Efficient data deduplication in Hadoop. LAP LAMBERT Academic Publishing.

[30] Prajapati, P., Patel, N., Macwan, R., Kachhiya, N., & Shah, P. (2014). Comparative analysis of DES, AES, RSA encryption algorithms. *International Journal of Engineering and Management Research, 4(1),* 132-134.

[31] Kolb, L., Thor, A., & Rahm, E. (2012). Dedoop: Efficient deduplication with Hadoop. *Proceedings of the VLDB Endowment* (pp. 1878-1881).

[32] Santos, W., Teixeira, T., Machado, C., Meira, W., Da Silva, A. S., Ferreira, D. R., & Guedes, D. (2007). A scalable parallel deduplication algorithm*. Proceedings of 19th International Symposium on Computer Architecture and High Performance Computing* (pp. 79-86).

[33] Kathpal, A., John, M., & Makkar, G. (2011). Distributed duplicate detection in post-process data de-duplication. *HiPC.*

[34] Patel, N., Shah, P., & Prajapati, P. (2015). Efficient data recovery with data integrity protection. LAP LAMBERT Academic Publishing.

Priteshkumar Prajapati obtained his bachelor's degree in information technology from Charotar Institute of Technology Changa, Gujarat Technological University, and Ahmedabad, India, in 2012. He had completed the M.Tech in information technology from Chandubhai S Patel Institute of Technology, Changa, CHARUSAT University in 2014. Currently he is working as Assistant Professor at the Department of Information & Technology, CHARUSAT, Changa, Gujarat. His research interests include big data, cyber security, data compression. He has published four Research Papers. Mr. Prajapati has ACM Membership since 2013. He secured Gold Medal in his M.Tech in information technology. He is also a reviewer of Journal of Big Data, Springer. He also reviewed couple of papers in 2nd International Conference on Bioinformatics and Computer Engineering (ICBCE 2016). He also reviewed couple of chapters in Book "Cryptography and Network Security: Principles and Practice, 7th Edition by William Stallings" and provided few technical suggestions.

Parth Shah is Professor in the Department of Information Technology at Charotar University Science & Technology, Changa, Anand, Gujarat. He graduated with a bachelor of engineering in computer engineering and post graduated in computer engineering and pursuing a PhD. His research interest includes information & network security, cloud computing and computer organization & architecture.

Amit Ganatra has received his BE and M.E. in 2000 and 2004 from DDIT-Nadiad, Gujarat. He has completed his Ph.D. in information fusion techniques in data mining from KSV University, Gandhinagar, Gujarat. He is a member of IEEE and CSI. He has 15+ years of teaching and research experience. He has published and contributed over 100+ papers. He is concurrently holding Professor, headship in the Computer Department and Deanship in Faculty of Technology-CHARUSAT, Gujarat.

Sandip Patel obtained his bachelor's degree in information technology from CSPIT Changa, Gujarat Technological University Ahmedabad in 2012. He had completed the M.Tech in computer engineering from CSPIT, Changa, CHARUSAT University in 2015. Currently he is working as Assistant Professor at the Department of Information & Technology, CHARUSAT, Changa, Gujarat. His research interest include cloud computing, networking and documentation in latex.