Object-Based Storage Accompanying File System For Improved Data Persistent And Processing Capabilities

Vandana Bhagat, Anshul Saxena, Amrita Tamang

Abstract: The world is leaning towards perceptive analytics at a rapid pace in order to improve business objectives at low cost and to achieve accuracy in the performance. It is the need of the hour to collect and analyse the data on a real-time basis. Incorporation of intelligent devices like sensors is available to collect the data, advanced software and devices to store the data and excellent tools to analyze the data in order to generate powerful reports and effective decision making. A significant amount of data needs to be saved and retrieved optimally to achieve the objective. Data warehouses present the solution to meet the significant data challenges. A framework like Hadoop is commonly used to meet for big data collection. Due to various challenges faced in implementing HDFS, industries are integrating object-based storage pattern for optimized data maintenance. There is a need to review HDFS and Object-based storage architecture and appropriate APIs to handle those operations. This research paper intends to introduce object-based storage services to store massive data in an optimized way and data access methods in real-time applications.

Index Terms: Block-Storage, File Storage, Hadoop, Hadoop Distributed File System (HDFS), Object-Based Storage, Amazon S3, OpenStack Swift.

1. INTRODUCTION

Newly evolved robotic data collection devices generate a massive amount of data every day. Many embedded applications and devices were collecting the data from multiple sources and converting that data into an insight is a next challenge. This data is present in structured as well as unstructured format. The analysts from IDC are expecting that by 2025, the global data sphere will grow to 163 zettabytes. That is an increase by more than 1000 % from the 16.1 ZB of data of 2016. [1]. Three types of applications are generating enormous data due to user-friendly applications, namely, healthcare, social media and e-commerce sites. The data generated and stored is mostly in the form of MP3, MP4 or images. More than this, many companies are storing and delivering data to their customers to meet their increasing demand [1]. It is collectively responsible for a vast amount of data generation. The real challenge in such case is to maintain this data for easy access which responds and gives required output in the fastest way. In the upcoming days most of the companies and businesses are planning to incorporate IoT technology as part of their product offering. IoT based applications is going to generate a significant amount of data which needs to be transferred and analyzed for each millisecond [2]. Due to this application-based approach, the demand for optimized data storage solutions has been increased to handle variety of digital contents.1 The solution for this is not only to generate and use significant data storage devices, but it needs intelligent data storage strategies for optimized use of memory [3]. Object-based storage pattern gives the ability to store the data as an object with its metadata, instead of the file system [3] where collectively maintained metadata generates bottleneck to data access [2]. Thus, objects can be regarded as convergence of two technologies: block and files [4][5][6][7]. The remainder of this paper is organized as follows: we have chronicled the evolution of object-based storage from disk to cloud in section1.2. In the subsequent sections we have discussed about the technologies which makes object-based storage ticks. Further the challenges faced; and how object-based storage system has helped in overcoming those challenges has been discussed. In the last section we have concluded with the discussion on how object-based storage system is playing a major role in deployment of cloud-based technology as well as in the system involved in the processing of unstructured data.

2 RELATED WORKS

Erik et al[8]. have found out that there is a decrease in the cost of processors and memory storage devices. Due to this storage system, designers are designing and utilizing excess computer power to perform more complex processing and optimizations. It is used to run database, data mining system and multimedia using objects-based storage on disks. They have introduced the system known as active disks that can help in running application-level code on individual disk drives. Ellard et al. [9] have compared the workload of contemporary email and research workloads through (Network File System) NFS traces. They have found a strong relationship between the name of file & its size, lifetime through out of order NFS calls. They have pointed out the need of high-performance devices which can overcome the limitation of disks storage space. Roselli et al. [10] have reported that because of the increasing gap between processor speed and disk latency file system performance is primarily determined by its disk behaviors. Since disk has limited storage space, Welis[11] has presented a distributed file system using object-based storage. They have leveraged device intelligence by using dynamic distributed metadata cluster. Sotomayor et al. [12] have reported the migration of memory power from disks to the cloud, also known as Infrastructure as a service (IaaS) through data centers as virtual machines using object-based storage system. Vozemedine et al. [13] have detailed and analyzed the role of a cloud operating system. They have found out that COS plays a vital role as part of modern technology which makes use of OBS for accessing and deploying virtual resources in remote cloud infrastructure. Nabi et al. [14] in their survey have analyzed and compared 100 paper from industry and academia. They have cited OBS as a critical part of cloud setup, which can help in overcoming the downtime and ensuring higher availability for the cloud services. Riley et al. [15] have analyzed and written that object-based storage has helped in overcoming three main challenges in cloud computing namely standard reference architecture, integration

with existing Infrastructure and process and maintenance & automation of the process. Tushti Narayani [16], in his research paper “Challenges with Hadoop “has discussed the need for data storage for growing business and has presented Hadoop as a solution for that. Though Hadoop is proved to be efficient for big data, many organizations are facing challenges in implementing Hadoop. These challenges might get solved with updated version of Hadoop ecosystem. Use of file system adds limitations. The author is convinced that despite challenges, Hadoop will serve as a perfect Big Data solution A White paper "OBJECT STORAGE ARCHITECTURE", has discussed the basic structure of Object-based storage. Object-based storage device and its data read/write operations give an alternate way to maintain the data efficiently. [18]Article form Apache software foundation, " Hadoop OpenStack Support: Swift Object Store" has described all fundamentals of Swift Object Store. OpenStack Swift is an open-source object-based storage service, which provides additional advantages over file system storage. [19].In an article written by Sunita Sharma "How Object Storage can improve Hadoop Performance", has listed the problems faced by HSFS Hadoop and how object-based storage can improve it. Object-based storage provides various benefits like Scalability, Cost, Accessibility, Durability and Elasticity. [20]

3 HDFS TECHNOLOGY

HDFS stores a large amount of data in the distributed format [27]. It is fundamentally blocked structured file system in which each file gets segregated into multiple blocks [28]. Due to being a name, distributed, these blocks get stored across numerous machines. HDFS follows a master/slave architecture pattern [19]. HDFS facilitates bulk data storage, and less intervention provides distributed and parallel processing simultaneously, helps to scale out, manages rollback and offers integrity by maintaining replicas of actual data. As shown in Fig 1, HDFS architecture is comprised of multiple compartments to store and manage the data.

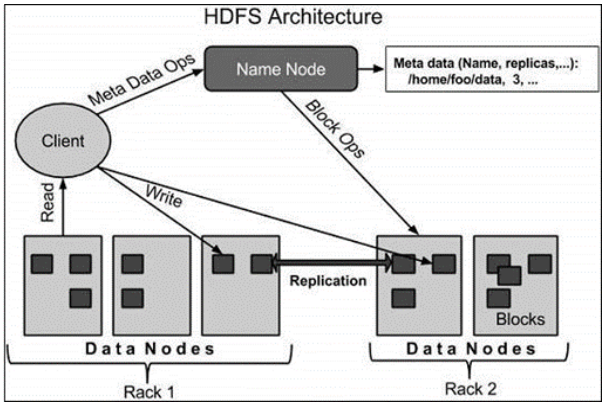

Fig. 1. *HDFS Architecture [21]*

1) Namenode: It works as a master node in the file system. It holds metadata of all the data in the system and regulates file access by the client. Basic file operations like file/directory opening, closing or renaming is handled by namespace.

2. Datanodes: It manages low level read/write requests on the files. It accomplishes block operations like create, delete and replicate.

3. Block: The distributed file segments get stored in individual data nodes which are nothing but blocks in HDFS. Block is the tiniest portion of the file which HDFS can read or write. [21]

To provide fault tolerance, HDFS maintains three copies of each block on different data node which gets physically distributed across the file system.

4 OBJECT-BASED STORAGE

Unlike a file system, object-based storage does not contain the complex hierarchical structure of directories and folders. As shown in Fig 2, it maintains a simple repository with its metadata, unique ID and actual data. As metadata gets stored with the object, it is highly customizable. Object-based storage eliminates challenges faced by a file system such as scalability, storage complexity and unstructured data storage with different attributes.

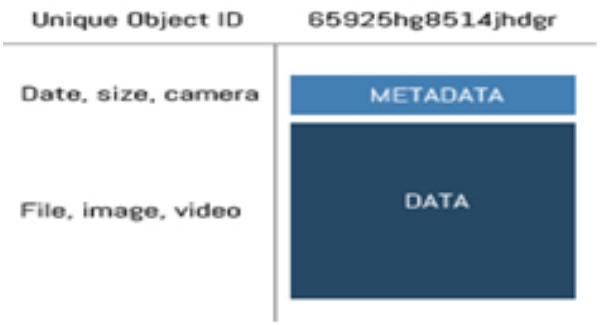

Fig. 2 *Object Based Storage Pattern [22]*

As metadata gets stored with the object, it is highly customizable. Object-based storage eliminates challenges faced by a file system such as scalability, storage complexity and unstructured data storage with different attributes. Object access the data through, Application Programming Interfaces (APIs). These APIs use RESTful web services to locate object anywhere via the Internet. Basic HTTP commands used by RESTful APIs are PUT/POST to insert a new object, GET to fetch the object and DELETE to remove the object [23].

5 HDFS CHALLENGES

1. Hadoop gets updated and evaluated continuously. Cons of this are, it is not stable at all, but pros is updates are always bringing novel models for computing and data storing capabilities. [20]
2. HDFS does not allow independent scaling. We cannot add one resource without the other.
3. As namenode is a starting point to reach rest of the nodes in Hadoop, if it gets fail, it becomes challenging to access the rest of the cluster. In such a case, Namenode becomes responsible for a single point of failure. [20].
4. HDFS does not do random reads very well.
5. To protect data, HDFS makes three copies of each dataset. Because these are enormous datasets, the costs of storing all those copies are high. The cost of adding more data capacity gets compared against the cost of saving all those copies when only 1/3 of the data [24].
6. HDFS follows master/slave architecture. If the master node fails, the rest clusters cannot be accessed. This cause a single point of failure.
7. In HDFS, compute power and storage capacity should

run together to insert a new node. When an entity needs more storage, it must also add a comparable amount of processing power and vice versa. One cannot add one without the other.[24]

8. Hadoop, being filesystem follows schema-less architecture. Data redundancy is possible due to the absence of constraints. Developers needs to optimize the data flow as optimization is not available.[25]
9. Business logic and infrastructure APIs have no clear separation, which further increases the burden upon developers.
10. There is absolutely no notion of transaction consistency or recovery checkpoints because this was designed to be a filesystem.
11. HDFS requires specialized expertise to use it effectively.
12. Hadoop does not have essential SQL functions like 'group by' analytics, subqueries and further [25].

6 OBJECT - ORIENTED STORAGE AS A SOLUTION

1. Object-based storage is easily scalable beyond Petabytes by adding a node. Data accessibility and processing becomes relaxed task when it gets stored as an object.[20]
2. Data loss is very well handled by object-based storage with the use of erasure coding. Substitute data can be accessed from another instance of the object if one instance fails.
3. Cheaper data storage is possible using object storage and gets accessed for in-memory for processing.
4. In object-based storage, all the metadata gets stored with a unique ID. Because of this, it does not depend on any master node to finish its process.[3]
5. The flat namespace organization of the data along with its expandable metadata functionality allows for capacity and compute to be scaled independently. As need changes, one can add object storage nodes, without having to add computing power.[20]

7 OBJECT-BASED STORAGE PROVIDERS

Many applications use object-based storage for smooth unstructured data storage. It avoids lots of constraints which was faced by a file system. OpenStack Swift, Amazon S3 and many more applications provide object-based storage service. Though they both follow the same broad storage pattern, the architecture they use to store and access the data differs. It is important to know characteristics, data storage architecture and APIs to handle the requests and response with clients, before selecting appropriate data storage service. The following section explicates about two popularly used object-based storage applications.

a. OpenStack Swift

OpenStack Swift is an open-source application where open source community updates the features regularly. Object-Based storage uses the hash value to locate the targeted object. Being object-based storage, swift stores multiple copies of data which gets accessed in case of any database damage. Fig. 3 shows components which takes part in the data storage procedure in OpenStack Swift.[23],[26] & [27].

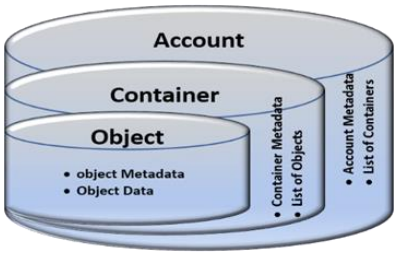

Fig.3 *OpenStack Data Storage Components [26]*

Account: Account is a storage area which is uniquely named. Account server contains the metadata and handles all the requests related to it. Each account store and maintains a list of containers. OpenStack Swift maintains this information in SQLite databases in the disk. Container: Container server handles requests related to the containers. It contains its metadata which holds details about container stored as SQLite database. Each container comprises a list of various objects. Objects: Object storage manages the real storage of actual data in the node. Objects store the data in the form of binary files which gets accessed through a specific path. It does not store information about its location, but it contains the information about the container to which it belongs. Objects maintain data and metadata together as one unit. Other than this proxy layer and consistency servers handle communication of data with the outside world and maintenance of data consistency.

Proxy: Proxy Layer is the layer which faces to the client and handles API request. Proxy layer uses standard HTTP verbs for requests and response codes. Proxy decides at which location the request gets sent. This layer handles various locations of data because swift keeps three complete copies of data to maintain the durability of data. Consistency: Data consistency gets managed with two processes, namely auditor and replicator. Individual auditor for an account, container and object continuously audit to detect any corruption in the file. Replicator checks for the similar updated data on each copy of an object. If it finds missing data, it brings it up-to-date. In order to handle operations in OpenStack Swift components, different APIs gets used. APIs are the requests and responses which help to transfer the data between different services. Table 1 contains a list of APIs for Accounts; Table 2 contains a list of APIs for Containers and Table.3 contains a list of APIs for Objects.[28]

Table 1: *API Requests for Accounts [28]*

Account	
GET/V1/{account}	Show account details and list conta
POST /v1/{account}	Create, update, or delete metadata
HEAD /v1/{account}	Show account metadata

Table 2: *API Requests for Containers [28]*

Containers	
GET /v1/{account}/{container}	Show container details and list objects
POST /v1/{account}/{container}	Create, update, or delete container metadata
HEAD /v1/{account}/{container}	Show container metadata
DELETE /v1/{account}/{container}	Delete container
DELETE /v1/{account}/{container}	Delete container

Table 3: *API Requests for Objects [28]*

Objects	
GET /v1/{account}/{container}/{object}	Get object content and metadata
PUT /v1/{account}/{container}/{object}	Create or replace object
COPY /v1/{account}/{container}/{object}	Copy object
DELETE /v1/{account}/{container}/{object}	Delete object
HEAD /v1/{account}/{container}/{object}	Show object metadata
POST /v1/{account}/{container}/{object}	Create or update object metadata

b. Amazon S3

Amazon S3 is closed source implementation. A single organization handles all the APIs. It contains well defined REST APIs which handles basic operations of buckets and objects. Fig. 4 shows components which takes part in the data storage process in Amazon S3.[29],[30],[31]

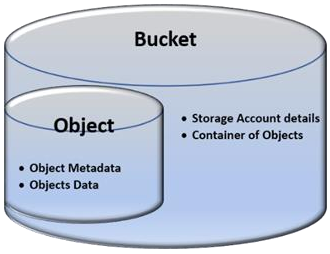

Fig.4 *Amazon S3 Data Storage Components [31]*

Bucket: Objects in Amazon S3 get stored in the bucket. Objects get accessed through the path, which also includes a respective bucket. Bucket servers play several roles like to manage account which is responsible for the storage, managing the access and generating the report. Objects: Object is the fundamental unit of object-based storage pattern. In Amazon S3, it stores actual data and metadata of the data. Metadata gets stored as key-value pair which stores information about a specific object. Keys: Key in Amazon S3 is the unique identifier of an object in the bucket. Each object has a private key inside a bucket. The object can be accessed using a combination of the bucket, key and version ID, which is unique. Consistency: Data Consistency gets maintained by keeping multiple (Generally 3) replicas of actual data. Data deletion or modification records gets maintained on each copy. New object creation, object data modification or object deletion takes place with appropriate consistency without disturbing any of other processor data in the object. Authentication in S3 gets managed by Access key and secret access key which has algorithms to handle each incoming request for its authentication.[32],[28],[27],[31]. To handle operations in Amazon S3 components, different APIs are used. APIs are the requests and responses which help to transfer the data between different services. Table 4 contains list of APIs for Bucket and Table 5 contains list of APIs for Objects.

Table 4: *API Requests for Bucket [28]*

Bucket	
PUT Bucket	creates a new bucket
GET Bucket (List Objects)	returns some or all (up to 1,000) of the objects in a bucket.
DELETE Bucket	deletes the bucket named in the URI.

Table 5: *API Requests for Object [28]*

Objects	
PUT Object	adds an object to a bucket.
PUT Object -Copy	creates a copy of an object that is already stored
GET Object	retrieves objects from Amazon S3
Delete Multiple Objects	delete multiple objects from a bucket using a single HTTP request.
DELETE Object	removes the null version (if there is one) of an object
HEAD Object	retrieves metadata from an object without returning the object itself.

8 TYPICAL APPLICATIONS OF OBJECT-BASED STORAGE

Disaster recovery: Backup and archiving: Object storage is an ideal alternative solution for data replication, given its easily-deployed network-based accessibility. It can be used either by replacing the disaster recovery site with Object Storage or by Replacing the complete backup solution with Object Storage.

· Static website hosting: One of the critical advantages of Object Storage is the ability to distribute data objects directly to the Internet using HTTP. Object Storage allows to create and modify a static website directly from the graphical user interface, or via API requests. Static files like PDF documents, zip archives, and video files generated through e-commerce or enterprise website consumes lots of storage cost of the server. Object Storage is an ideal solution to offload static content to a resilient and always-on platform that can serve files directly to customers, via free Internet bandwidth.[2] Document Store and file-sharing: Maintaining a server estate and storage platform for a document management system is cumbersome. Transforming the storage backend entirely to an Object Storage platform is a logical step to lower the administrative burden. Big Data analytics: For big data, the biggest problem is not data, but the metadata which contains information necessary for analysis and makes the data more valuable. Object Storage supports the creation of user-defined object metadata (key-value pairs) together with a variety of system-defined metadata that is stored by default. [2] Object storage is getting implemented in healthcare, financial and entertainment industries. It is now becoming attractive to an even broader group of industries such as obedience, governance, and archival requirements which drive the demand for enormous data fountains within their IT organizations.[4].Especially in

healthcare sector where due to large number and variety of transactions object- based storage is providing the storage to the data archive problems[33]. According to the use case presented by western digital, 90% of top 50 banks are using the advanced analytics solutions which can be delivered on the analysis of live streaming of data. To achieve this goal they deploy object-based storage to run the critical data mining programs.

9 ADVANTAGES OF DEPLOYING CLOUD BASED OBJECT STORAGE SYSTEM

1. Flexible storage class tiers and low-cost archive - Object-based storage provides options of storing the legacy data at a lesser cost.
2. Higher durability and resiliency - Performance and reliability can be derived from OBS based cloud storage system.
3. Data protection and security - Using encryption-based security and state-of-art of firewall cloud-based OBS helps in securing the data.
4. Fast data transfer - High-speed data transfer has got achieved with the help of cloud-based OBS.
5. Query data in place - Using Cloud OBS data can be tapped to answer different queries without setting complex ETL process.
6. Immutable object storage - Various records about customers, as well as stakeholders, can be stored using OBS.

Object Expiration - Cluttering of storage space can be managed well with timely deletion of unwanted and unused data. [23]

10 CONCLUSIONS

Due to IoT portable devices, the significant amount of structured and unstructured data is getting generated. Such big data gets maintained in data warehouses. Data storage is the main challenge in front of all industries providing data warehouse and analysis tools for decision making. There are enough solutions for structured data, but still, researchers and industry people are struggling for optimized storage of unstructured data. A typical data storage for the warehouse is the HDFS system. File system performed efficiently when the data was structured and limited, but with growing data, it started facing many problems. Many researchers and data storage industries came up with object-based storage as an optimized solution. Object-based storage is different from traditional file data storage and block data storage, could address many problems faced by the file storage system, but it was lagging for some solutions which could be fulfilled by HDFS. Apache came up with combined advantages of HDFS and Object-based storage named as Hadoop OpenStack Support: Swift Object Store. It was a new way of data storage and getting used by different industries. Amazon S3 also provides object-based storage service like OpenStack swift. Though they differ in data storage and data access strategies, they both follow similar object-based storage. The comparative performance of different types of storage is essential to study. Theoretically, a combination of object-based storage and file system storage makes sense, but its applications and performance need to be measured. Data storage need will be never-ending. Researchers will always be in search of the more and more optimized way of data storage and query execution as per the need of ever-changing IT services.

REFERENCES

[1] Michael Nuncic, "The Evolution of Storage: File Storage vs Block Storage vs Object Storage – Part 1.", 2018
[2] Interoute is not GTT, "Object Storage Use Cases."
[3] Margaret Rouse, "object storage.", 2017
[4] Ingo Fuchs, NetApp, "Check Your Mirror: Object Storage May Be Closer than It Appears"
[5] Nitish Tiwari, "Object storage: What is it all about?", 2016
[6] Nitish Tiwari, "Object storage in practice: Creating a reliable data store.", 2016
[7] Oracle and/or its affiliates, "Overview of Object Storage.",2019
[8] E. Riedel, G. Gibson, and C. Faloutsos, “Active Storage for Large-Scale Data Mining and Multimedia Applications,” Int’l. Conf. Very Large DBs, New York, NY, Aug. 24–27, pp. 62–73, 1998
[9] Ellard, Daniel, Jonathan Ledlie, Pia Malkani, and Margo Seltzer, “Passive NFS tracing of email and research workloads. Proceedings of the Second Symposium on File and Storage Technologies”, San Francisco, CA, 203-216. Berkeley, California: USENIX Association., 2003
[10] D. Roselli, J. R. Lorch, and T. E. Anderson, “A Comparisonof File System Workloads,” USENIX Annual Tech.Conf., San Diego, CA, June 18–23, pp. 41–54, 2000
[11] S. A. WeilS, A. Brandt, E. L. Miller, D. D. E. Long, C. Maltzahn, “Ceph: a Scalable, High-performance Distributed File System”, OSDI '06 Proceedings of the 7th Symposium on Operating Systems Design and Implem
[12] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster. “Virtual Infrastructure Management in Private and Hybrid Clouds. Internet Computing,” IEEE Internet Computing, 13(5):14-22, 2010.
[13] R. Moreno-Vozmediano, R. S. Montero, I. M. Llorente. “IaaS Cloud Architecture: From Virtualized Data Centers to Federated Cloud Infrastructures,” IEEE Computer, 45(12):65-72, 2012.
[14] M. Nabi, M. Toeroe, F. Khendek. “Availability in the cloud: State of the art” Journal of Network and Computer Applications, Vol. 60, pp. 54-67, 2016
[15] J. Riley, J. Noss, J. Cuff, I. M. Llorente, “A HighAvailability Cloud for Research Computing”, IEEE Computer, pp: 91-95, Issue No. 06 - June (2017 vol. 50)
[16] Tushti Naryani (IJARCST 2015), "Challenges with Hadoop", International Journal of Advanced Research in Computer Science & Technology
[17] Panasas, White paper, "OBJECT STORAGE ARCHITECTURE.", 2007
[18] The Apache Hadoop Foundation, "Hadoop OpenStack Support: Swift Object Store.", 2018.
[19] Shandor1280, “What is HDFS” , intellipaat.com, (blog: https://intellipaat.com/blog/what-is-hdfs/), 2016
[20] Sunita Sharma, "How Object Storage can improve Hadoop Performance.", 2018
[21] Tutorial points, “Hadoop - HDFS Overview”, (source: https://www.tutorialspoint.com/hadoop/hadoop_hdfs_overview.html) 2019
[22] Qstar Technologies, “Object Storage Technology-Overview”, (source:]https://www.qstar.com/qstar-object-storage/), 2019
[23] IBM Cloud Education, “Object Storage”, (source: https://www.ibm.com/cloud/learn/object-storage), 2019

[24] Mary Ann Richardson, One year ago, "3 reasons to replace HDFS with Object Storage and one reason not to."
[25] Tushti Narayani,, "Challenges with Hadoop", IJARCST - 2015
[26] L. Rupprecht, R. Zhang , B. Owen, P. Pietzuch , D. Hildebrand, “SwiftAnalytics: Optimizing Object Storage for Big Data Analytics”, 2017.
[27] SwiftStack, “OpenStack Swift Architecture”, (source: https://www.swiftstack.com/docs/introduction/openstack_swift.html”)
[28] Ankit Agrawal, “How to ensure OpenStack Swift & Amazon S3 Conformance for storage products & services supporting multiple Object APIs”, Tata Consultancy Services Ltd, 2017
[29] Amazon Web Services, “Amazon Simple Storage Service: Getting Started Guide”, 2019
[30] Reynold Xin, Josh Rosen and Kyle Pistor, “Top 5 Reasons for Choosing S3 over HDFS”, databricks , (blog: https://databricks.com/blog/2017/05/31/top-5-reasons-for-choosing-s3-over-hdfs.html), 2017.
[31] Amazon Web Services, Inc , “Introduction to Amazon S3”, Amazon Simple Storage Service, (source: https://docs.aws.amazon.com/AmazonS3/latest/dev/Introduction.html), 2019
[32] Dean Hildebrand, Bill Owen, “An Overview of OnPremise File and Object Storage Access Protocols”, IBM
[33] Whitepaper, Caringo, Inc., “Object Storage for Healthcare”, SOLUTION BRIEF—HEALTHCARE 2016.