3. Storage in Cloud

3.1 Storage system architecture

(a) Explain Storage system architecture.

Ans. Storage system architecture:

I. The design and arrangement of data storage resources and services inside a cloud computing environment is known as storage system architecture.

II. Scalable, dependable, and cost-effectiveness are all features of cloud storage infrastructures.

III. They frequently use a variety of technologies and methods to accomplish their objectives.

IV. The most important information for the firm is protected by specialized settings called storage system architecture.

V. It manages and stores a lot of data, necessitating specialized solutions with higher dependability and manageability.

Public API's for Data & Management, Object Storage, Virtual Compute Server, Block,file, object Storage, Logical storage pools, physical Storage Server, Cloud Storage Location.

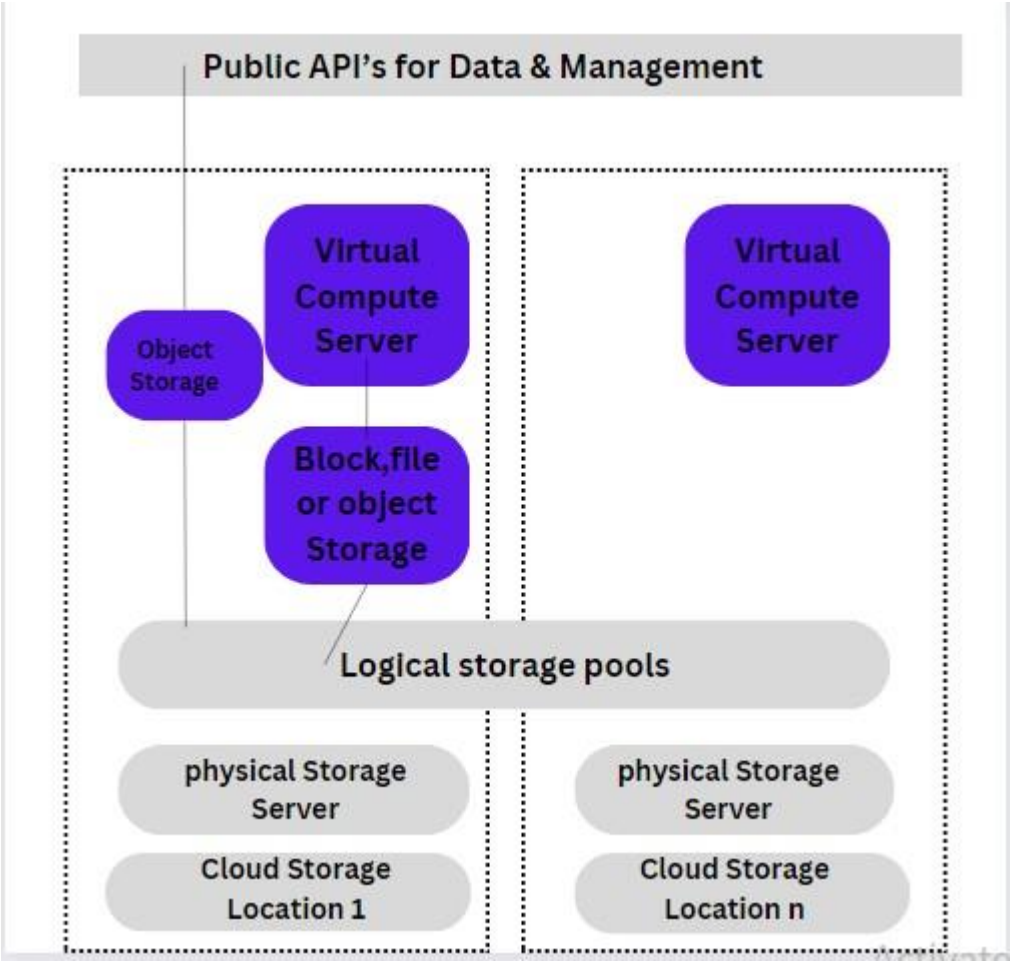

Figure.: Storage System Architecture in cloud

Components of Storage System Architecture in cloud: i)

Public API's data for Data and Management:

1. Software applications can interact and communicate with external services, libraries, or platforms using public APIs (application programming interfaces).
2. There are a number of APIs available from different cloud service providers and storage services that let you manage and interact with data in the context of storage systems.

ii) Virtualized Compute Server:

1. A server that has been partitioned into numerous virtual machines (VMs) using virtualization technology is referred to as a virtualized compute server.
2. With the help of this technology, numerous isolated instances of operating systems and programs can run on a single physical server, each of which functions as if it were on a separate dedicated server.
3. Numerous advantages of virtualization include better resource usage, flexibility, scalability, and management simplicity.

ii) Object Storage:

1. Object storage is a type of data storage architecture that manages and organizes data as discrete, self-contained objects rather than as blocks or files. Each object contains the data itself, metadata describing the data, and a unique identifier. Object storage systems are designed to provide scalable, highly available, and cost-effective storage for a wide range of applications and use cases.
2. Many cloud providers offer object storage services, such as Amazon S3, Google Cloud Storage, and Azure Blob Storage. These systems store data as objects with unique identifiers (such as URLs) and provide features like scalability, data durability, and cost-effective storage.

iv) Block Storage:

1. The name originally referred to a hosted object storage service, but it has now expanded to cover additional forms of data storage that are also offered as services, such as block storage.
2. Block Storage constitutes data in data node & name node format. **v) File Storage:**

1. Cloud can offer you the possibility of storing your files and accessing, storing and retrieving them from any web-enabled interface. The web services interfaces are usually simple. At any time and place you have high availability, speed, scalability and security for your environment.
2. In this scenario, organizations are only paying for the amount of storage they are actually consuming, and do so without the worries of overseeing the daily maintenance of the storage infrastructure.

vi) Logical Storage Pool:

1. A logical storage pool is a virtualized storage resource that portrays physical storage devices as a single, simple-to-manage storage area by abstracting them from one another.
2. It is a fundamental idea in contemporary storage system architecture and is frequently applied to both conventional on-premises storage systems and cloud computing platforms. Greater flexibility, scalability, and easier management of storage resources are the aims of a logical storage pool.

vii) Physical Storage Server:

1. A physical storage server is a specialized piece of hardware created to offer storage space for data, programs, and services.
2. It is a crucial part of the architecture of storage systems and can perform a variety of tasks, from simple file storage to more complex ones like hosting databases, virtualization environments, and cloud storage services.

viii) Cloud Service Location

1. "Cloud service location" typically refers to the geographical data center or data centers where a specific cloud service or application is hosted and served from within a cloud computing provider's infrastructure. 2. Major cloud providers like Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP) have data centers in various regions around the world, and users can select the appropriate region based on their requirements.

3.2) Virtualize Data Centre(VDC) :Architecture, VDC Environment, server,storage, networking VDC:

1. In order to build a more adaptable, scalable, and effective environment, an IT architecture known as a virtualized data center.
2. It uses virtualization technologies to abstract and pool computing, storage, and networking resources.
3. Physical hardware resources are abstracted and managed as virtual resources in a virtualized data center, which improves resource usage, management efficiency, and agility.
4. Organizations may now manage IT resources more effectively and quickly while spending less on hardware because of virtualized data centers.
5. They give firms the foundation they need to adapt to the demands of quickly shifting technological environments.

VDC Architecture:

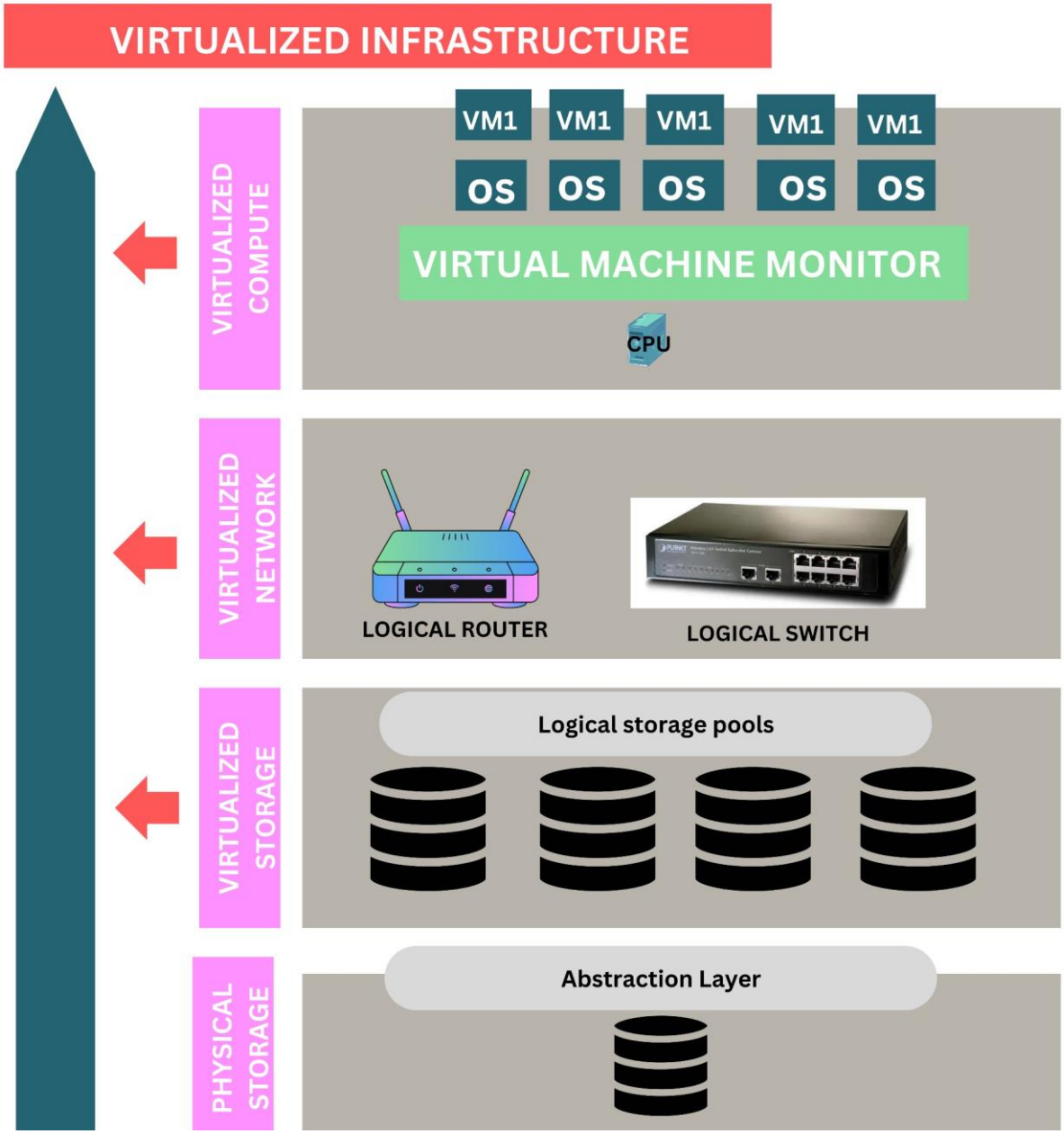

FIGURE: Virtualized Data Center Architecture

There are four phases to achieve VDC:

Phase1: Virtualized Infrastructure

1. Virtualized infrastructure abstracts and pools computing, storage, and networking resources. It utilizes technologies like server, storage, and network virtualization to create flexible, scalable, and efficient environments.
2. Virtual machines run on shared physical servers, optimizing resource utilization. This approach enables easy management, rapid provisioning, and improved resource allocation. **Phase 2: Virtualized Compute**

    1. Virtualized compute abstracts physical servers into virtual machines (VMs), enabling multiple OS instances on one server. Hypervisors manage VMs, optimizing resource usage and isolation.
    2. Scalable and flexible, it accelerates provisioning and adapts to varying workloads. Virtualized compute enhances resource utilization, simplifies management, and aids in cloud deployments.

Phase 3: Virtualized Network

1. Virtualized networking abstracts and segments networks into virtual networks, decoupling from physical hardware. Software-defined networking (SDN) controls and manages these virtual networks centrally.
2. It enhances network flexibility, security, and efficiency, enabling dynamic configurations. Virtualized networks are key to modern data centers and cloud infrastructures. **Phase 4: Virtualized Storage**

    1. Virtualized storage abstracts physical storage resources into virtual pools, managed centrally. It optimizes storage utilization, allowing dynamic allocation and scaling.
    2. Different tiers of storage cater to varying performance and cost needs. Virtualized storage enhances data management, disaster recovery, and supports cloud services.

3.3 Block and file level storage virtualization, Virtual Provisioning, and automated storage tiering:

Types of Storage Virtualization:

1. **Block level storage virtualization**
2. **File level storage virtualization Block level storage**

virtualization:

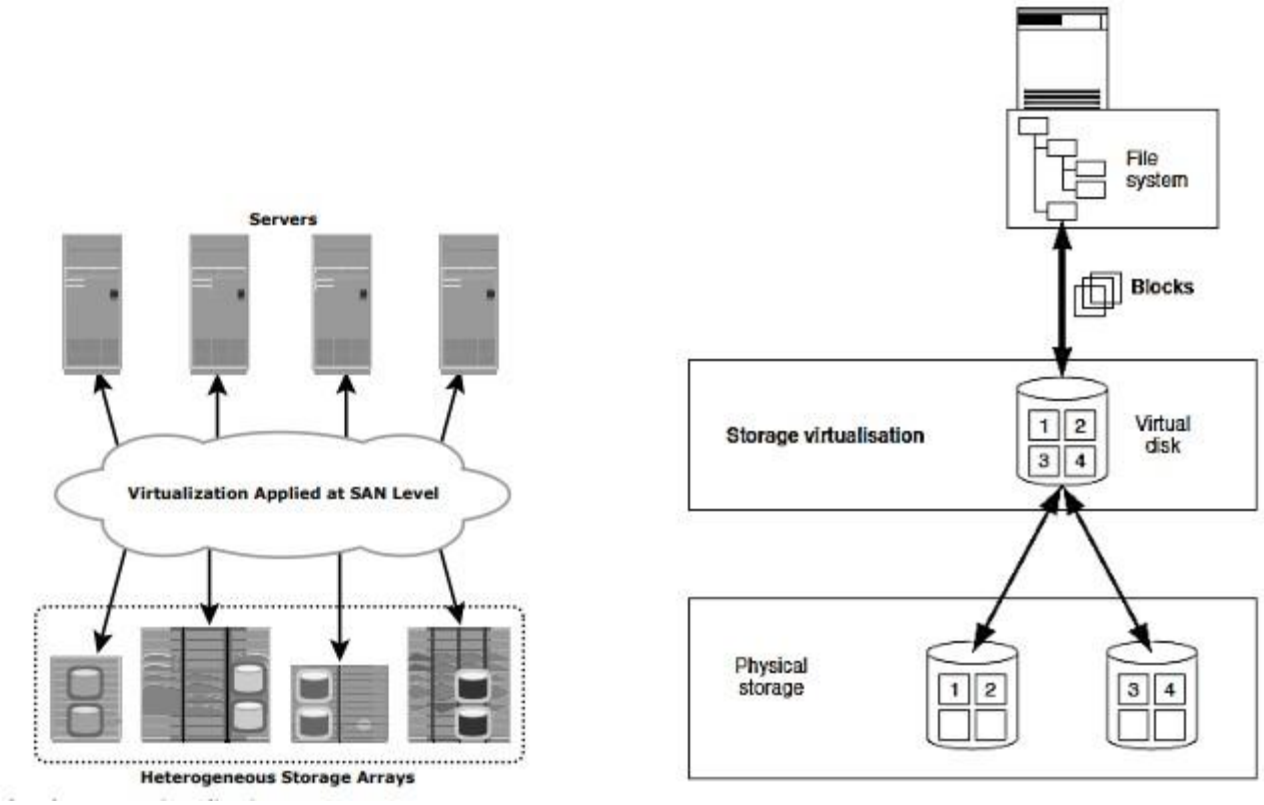

Figure: Block-level storage virtualization

1. Block-level storage virtualization provides a translation layer in the SAN, between the hosts and the storage arrays, as shown in Figure.
2. Block-level storage virtualization extends storage volumes online, resolves application growth requirements, consolidates heterogeneous storage arrays, and enables transparent volume access
3. Virtualisation on block level means that storage capacity is made available to the operating system or the applications in the form of virtual disks
4. In virtualisation on block level the task of file system management is the responsibility of the operating system or the applications
5. The task of the virtualisation entity is to map these virtual blocks to the physical blocks of the real storage devices.
6. In this, No physical changes are required because the host still points to the same virtual targets on the virtualization device. However, the mappings on the virtualization device should be changed. These changes can be executed dynamically and are transparent to the end user.
7. Deploying heterogeneous arrays in a virtualized environment facilitates an information lifecycle management (ILM) strategy, enabling significant cost and resource optimization.
8. Low-value data can be migrated from high- to low-performance arrays or disks. Detailed implementation of functionality and operation of block-level storage virtualization.

File level storage virtualization:

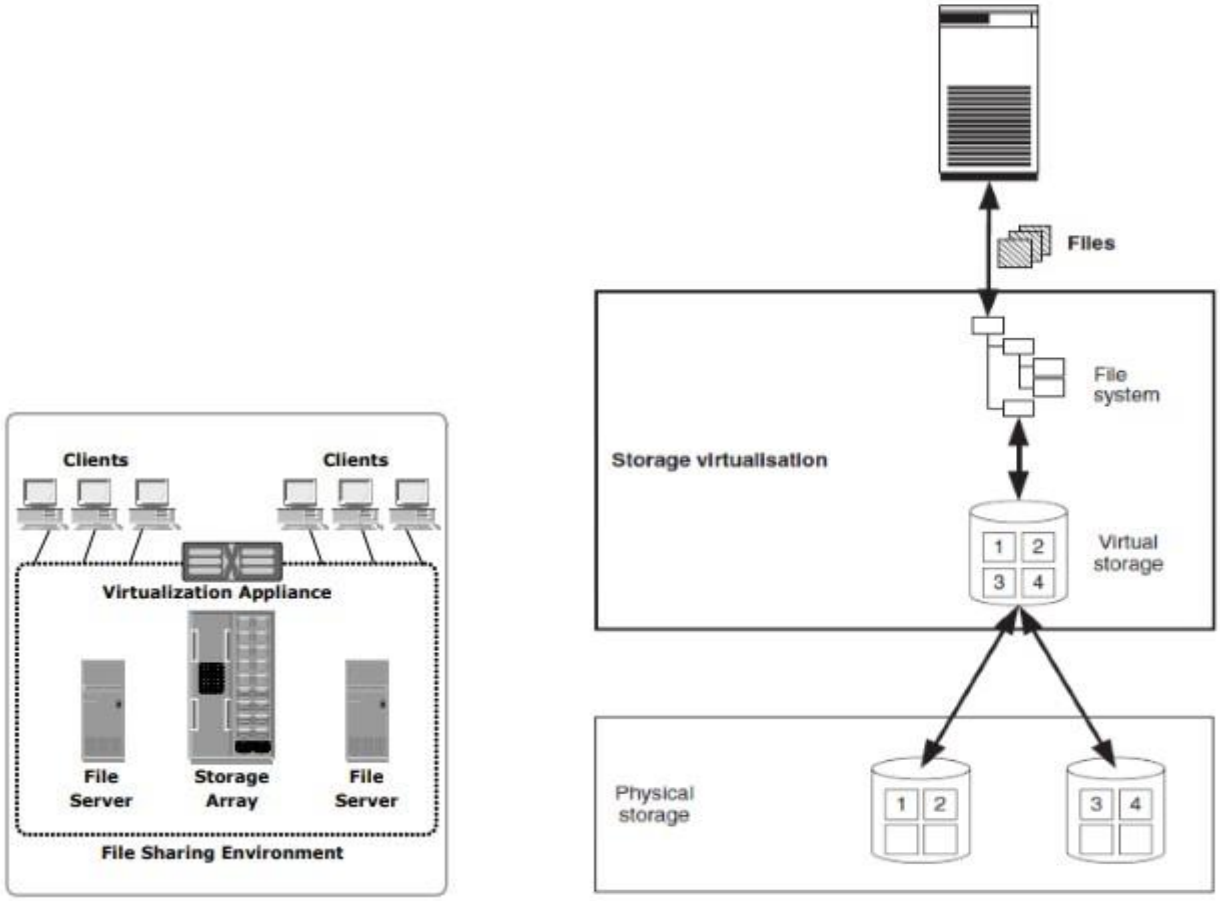

Figure: File Level Virtualization

1. File-level virtualization addresses the NAS challenges by eliminating the dependencies between the data accessed at the file level and the location where the files are physically stored. This provides opportunities to optimize storage utilization and server consolidation and to perform nondisruptive file migrations.
2. File-level virtualization simplifies file mobility. It provides user or application independence from the location where the files are stored. File-level virtualization creates a logical pool of storage, enabling users to use a logical path, rather than a physical path, to access files.
3. File-level virtualization facilitates the movement of file systems across the online file servers. This means that while the files are being moved, clients can access their files nondisruptively.
4. Clients can also read their files from the old location and write them back to the new location without realizing that the physical location has changed.
5. Multiple clients connected to multiple servers can perform online movement of their files to optimize utilization of their resources. A global namespace can be used to map the logical path of a file to the physical path names. 6. Virtualisation on file level means that the virtualisation entity provides virtual storage to the operating systems or applications in the form of files and directories. The physical blocks are presented in the form of a virtual file system and not in the form of virtual blocks.
7. The applications work with files instead of blocks and the conversion of the files to virtual blocks is performed by the virtualisation entity itself(This means, the task of file system management is performed by the virtualisation entity, unlike in block level which is done by OS or application).

Virtual Provisioning:

1. Virtual provisioning, sometimes referred to as thin provisioning, is a storage management strategy that enables numerous applications or systems to share a single pool of physical storage resources in order to more effectively distribute storage space.

2. The ability to deliver a LUN to a compute system with more capacity than what is physically allotted to the LUN on the storage array is known as virtual provisioning. It can be used in storage layer and compute layer implementations.
3. Physical storage is only allocated when the computation requires it, and provisioning decisions are not constrained by the amount of storage that is currently available.
4. In cloud environments, where resource optimization and cost effectiveness are crucial, virtual provisioning is frequently employed.

Automated storage tiering:

1. Storage Tiering prioritizes storage blocks into different categories, referred to as storage tiers, which provide various levels of performance and capacity based on price/performance considerations, performance/ bandwidth demands, frequency of use, and other criteria.
2. Storage Tiering enables users to flexibly assign applications to tiers with different drive types and RAID levels.
3. Infortrend's Automated Storage Tiering provides an architecture that fully consolidates the advantages of different storage media, including SSDs for high performance and near-line serial attach SCSI (NL-SAS) drives for storage capacity.
4. It helps users more easily accommodate and meet different service level requirements via easy-to-use GUI-based management tools.

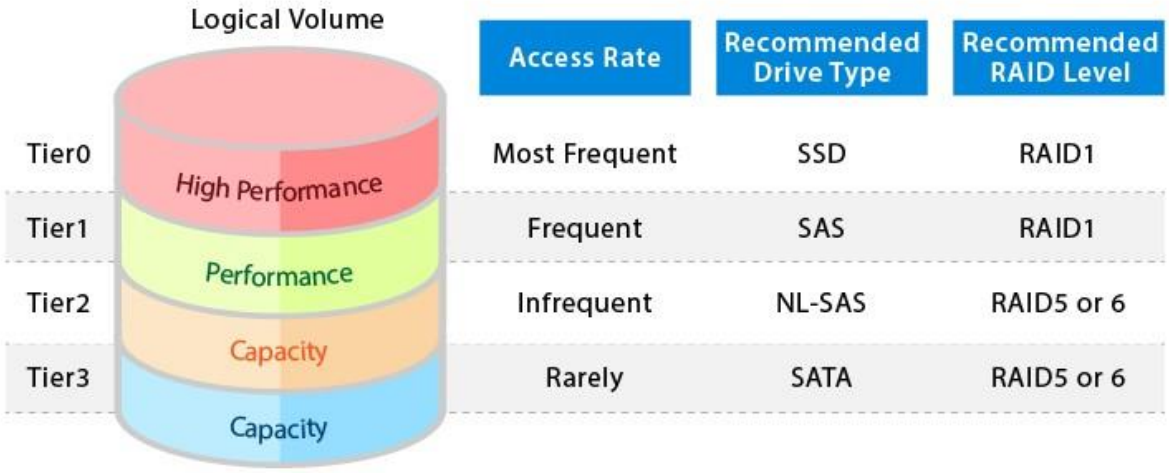

Figure: Automated storage tiering

3.4 Virtual Storage Area Network(VSAN) and benefits:

I. VSAN (Virtual Storage Area Network) is a storage solution that is used to create and manage storage for virtual machines. It is intended for usage in scenarios that leverage cloud computing, especially with virtualized infrastructure like VMware vSphere.

II. Centralized storage management is offered by VSAN for virtual machines and applications running in a virtualized environment.

III. Storage resources from various physical servers can be combined and shown as a single, shared storage pool using VSAN.

IV. The virtual machine disc files (VMDKs) and other data can then be kept in this pool. VSAN dynamically allocates storage for virtual machines according to requirements.

V. It uses distributed architecture and enables IT organizations to pool storage resources and dynamically provide storage to virtual machines on an as-needed basis, VSAN is especially well-suited for cloud computing environments.

VI. This can make it simpler to administer virtual environments and lower the expense of obtaining and maintaining real storage.

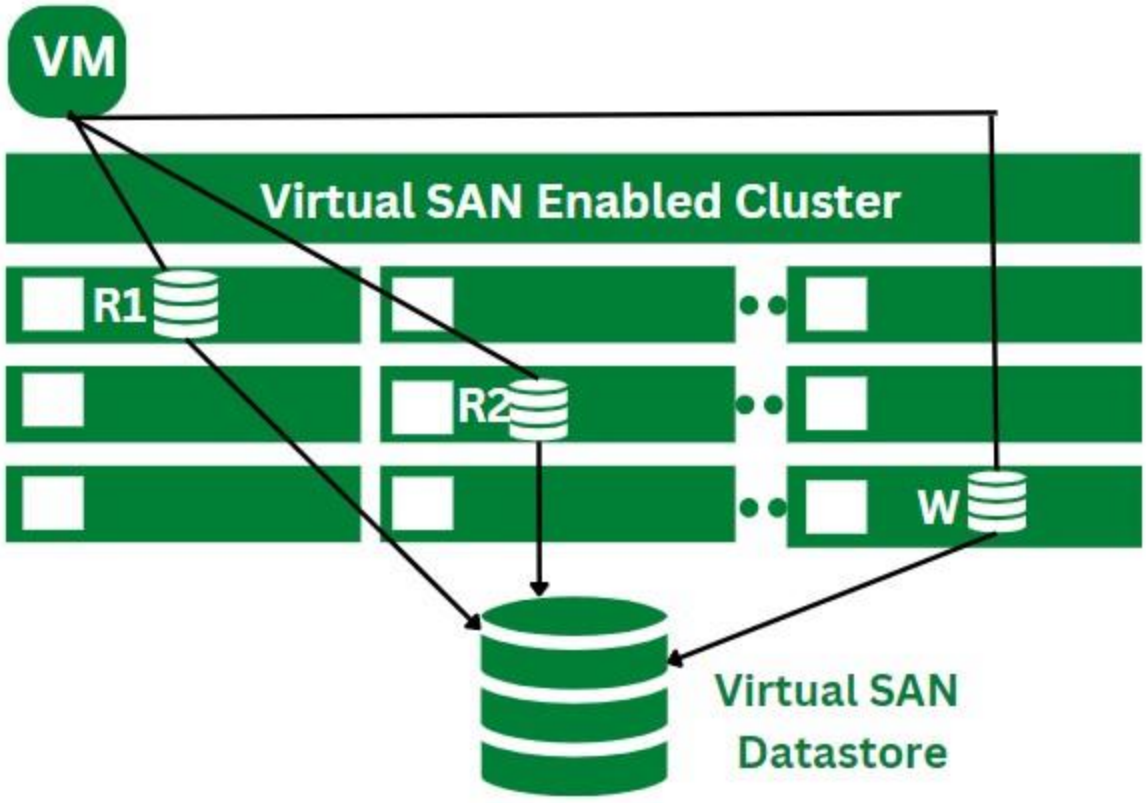

Figure: Virtual Storage Area Network Architecture

Applications of VSAN

1. Hybrid Cloud Storage: VSAN can be used to create a hybrid cloud environment where data can be stored and managed on-premises and in the cloud. Hybrid cloud storage
2. Virtual Desktop Infrastructure (VDI): VSAN can be used to offer storage for virtual desktop environments (VDI), enabling effective virtual desktop management and storage in the cloud.
3. Disaster Recovery and Business Continuity: Data can be copied to a secondary site in the cloud for protection against outages and data loss using VSAN to develop disaster recovery and business continuity solutions.
4. Application Development and Testing: VSAN can be used to offer storage for environments used for developing and testing applications, allowing programmers to build and test cloud-based apps.
5. Backup and Archiving: Storage for backup and archiving solutions can be provided by VSAN, allowing businesses to store and safeguard their data in the cloud.

Benefits of VSAN

1. Cost-effectiveness: VSAN does not require physical storage arrays therefore it is less expensive than conventional storage systems.

2. Scalability: VSAN is well-suited for cloud computing environments that demand the capacity to scale rapidly and efficiently. It can be simply scaled to meet changing storage requirements.
3. Increased performance: To deliver quick and dependable storage performance, VSAN makes use of the high-speed interconnects found inside the cloud computing architecture.
4. Flexibility: Block and file storage are both supported by VSAN, giving clients the option to select the type of storage that best suits their requirements.
5. Data security: VSAN has tools like data replication and snapshots that guard against data loss and guarantee that vital information is always accessible.
6. Simplified Management: Administration is made easier because of VSAN's integration with the VMware vSphere virtualization technology, which offers a single management panel.

3.5 Cloud file systems: GFS and HDFS, Comparisons among GFS and HDFS.

1. The Google File System (GFS) is designed to manage relatively large files using a very large distributed cluster of commodity servers connected by a high-speed network.
2. It is therefore designed to

   (a) expect and tolerate hardware failures, even during the reading or writing of an individual file (since files are expected to be very large) and

   (b) support parallel reads, writes and appends by multiple client programs. A common use case that is efficiently supported is that of many 'producers' appending to the same file in parallel, which is also being simultaneously read by many parallel 'consumers'.

3. As a result they also do not scale as well as data organizations built on GFS-like platforms such as the Google Datastore.
4. The Hadoop Dis- tributed File System (HDFS) is an open source implementation of the GFS architecture that is also available on the Amazon EC2 cloud platform; we refer to both GFS and HDFS as 'cloud file systems.'
5. The architecture of cloud file systems is illustrated in Figure.

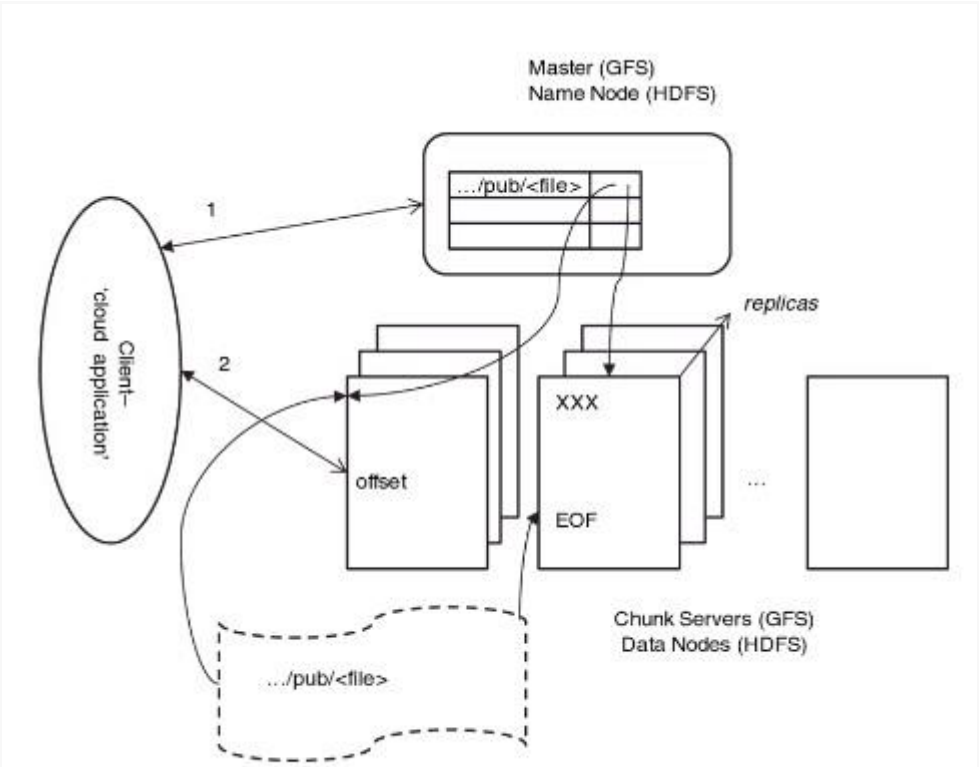

FIGURE 10.3. Cloud file systems

1. Large files are broken up into 'chunks' (GFS) or 'blocks' (HDFS), which are themselves large (64MB being typical). These chunks are stored on commodity (Linux) servers called Chunk Servers (GFS) or Data Nodes (HDFS); further each chunk is replicated at least three times, both on a different physical rack as well as a different network segment in anticipation of possible failures of these components apart from server failures.
2. When a client program ('Cloud Application') needs to read/write a file. It sends the full path and offer to the Master(GFS) which sends back meta-data for one (in the case of read) or all (in the case of write) of the replicas of the chunk where this data is to be found.
3. The client caches such meta-data so that it need not contact the Master each time. Thereafter the client directly reads data from the designated chunk server; this data is not cached since most reads are large and caching would only complicate writes.
4. In case of a write, in particular an append, the client sends only the data to be appended to all the chunk servers; when they all acknowledge receiving this data it informs a designated 'primary' chunk server, whose identity it receives (and also caches) from the Master.
5. The primary chunk server appends its copy of data into the chunk at an offset of its choice; note that this may be beyond the EOF to account for multiple writers who may be appending to this file simultaneously. The primary then forwards the request to all other replicas which in turn write the data at the same offset if possible or return a failure. In case of a failure the primary rewrites the data at possibly another offset and retries the process.
6. The Master maintains regular contact with each chunk server through heartbeat messages and in case it detects a failure its meta-data is updated to reflect this, and if required assigns a new primary for the chunks being served by a failed chunk server. Since clients cache meta-data, occasionally they will try to connect to failed chunk servers, in which case they update their meta-data from the master and retry.
7. It is shown that this architecture efficiently supports multiple parallel readers and writers. It also supports writing (appending) and reading the same file by parallel sets of writers and readers while maintaining a consistent view, i.e. each reader always sees the same data regardless of the replica it happens to read from.
8. Finally, note that computational processes (the 'client' applications above) run on the same set of servers that files are stored on. As a result, distributed programming systems, such as MapReduce, can often schedule tasks so that their data is found locally as far as possible, as illustrated by the Cluster system.

GFS Architecture:

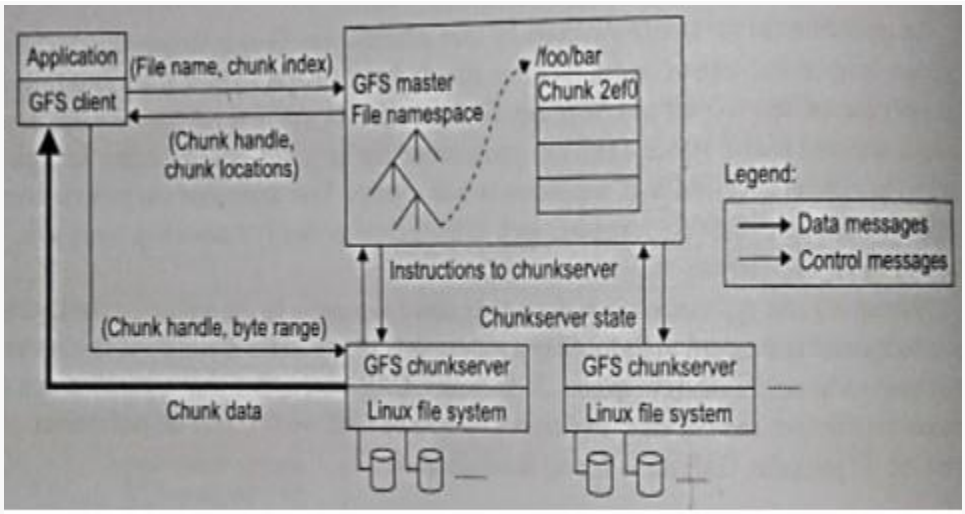

Fig. Architecture of Google File System (GFS)

1. There is single master in the whole cluster which stores metadata.
2. Other nodes act as the chunk servers for storing data.
3. The file system namespace and locking facilities are managed by master.
4. The master periodically communicates with the chunk servers to collect management information and give instruction to chunk servers to do work such as load balancing or fail recovery.
5. With a single master, many complicated distributed algorithms can be avoided and the design of the system can be simplified.
6. The single GFS master could be the performance bottleneck and single point of failure.
7. To mitigate this, Google uses a shadow master to replicate all the data on the master and the design guarantees that all data operations are transferred between the master and the clients and they can be cached for future use.
8. With the current quality of commodity servers, the single master can handle a cluster more than 1000 nodes.

The features of Google file system are as follows:

1. GFS was designed for high fault tolerance.
2. Master and chunk servers can be restarted in a few seconds and with such a fast recovery capability, the window of time in which data is unavailable can be greatly reduced.
3. Each chunk is replicated at least three places and can tolerate at least two data crashes for a single chunk of data.
4. The shadow master handles the failure of the GFS master.
5. For data integrity, GFS makes checksums on every 64KB block in each chunk.
6. GFS can achieve the goals of high availability, high performance and implementation.
7. It demonstrates how to support large scale processing workloads on commodity hardware designed to tolerate frequent component failures optimized for huge files that are mostly appended and read.

HDFS Architecture:

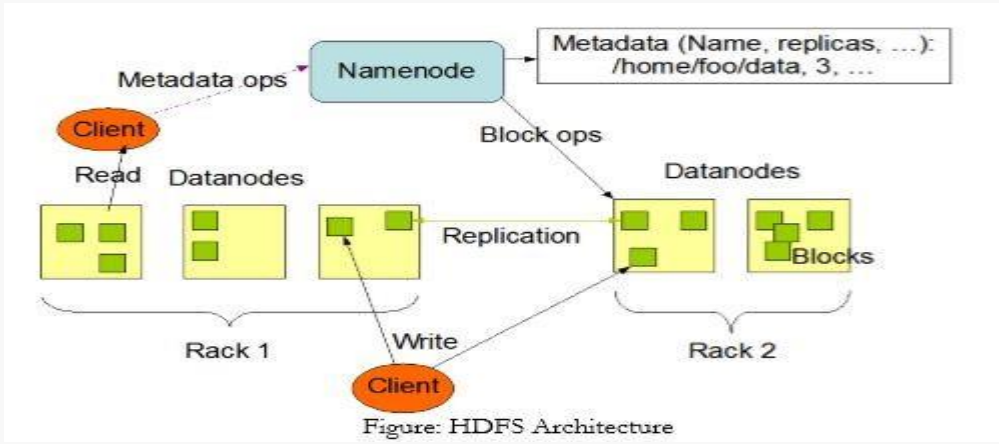

Figure: HDFS Architecture

1. The Hadoop Distributed File System (HDFS) is designed to provide a fault-tolerant file system designed to run on commodity hardware. The primary objective of HDFS is to store data reliably even in the presence of failures including Name Node failures, Data Node failures and network partitions.
2. HDFS uses a master/slave architecture in which one device (the master) controls one or more other devices (the slaves). The HDFS cluster consists of a single Name Node and a master server manages the file system namespace and regulates access to files.
3. **NameNode and DataNodes:** The NameNode and DataNode are pieces of software designed to run on commodity machines. These machines typically run a GNU/Linux operating system (OS). HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software.
4. **The File System Namespace:** The file system namespace hierarchy is similar to most other existing file systems; one can create and remove files, move a file from one directory to another, or rename a file. HDFS does not yet implement user quotas.The NameNode maintains the file system namespace.
5. **Data Replication:** HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file.
6. Block: Generally the user data is stored in the files of HDFS. The file in a file system will be divided into one or more segments and/or stored in individual data nodes. These file segments are called as blocks. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

Features of HDFS

1. Manages the files system manespare. Regulation clients access to files, It also execute file system operations such as naming, and opening files and directories data node.
2. Fault detection and recovery : Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanism for quick and automatic fault detection and recovery.
3. Huge data sets : HDFS should have hundreds of nodes per cluster to move the applications having huge data sets.
4. Hardware at data : A requested tasks can be done efficiently, when the computation takes place near the data. Especially where huge data base are involved it reduces the network traffic and increase the through PD. **Comparisons among GFS and HDFS:**

Key Point	HDFS Framework	GFS Framework
Objective	Main objective of HDFS to handle the Big-Data	Main objective of HDFS to handle the Big-Data
Language used to Develop	Java Language	C, CPP Language
Implemented by	Open source community, Yahoo, Facebook, IBM	Google
Platform	Work on Cross-platform	Work on Linux
License by	Apache	Proprietary or design by google for its own used.
Files Management	HDFS supports a traditional hierarchical directories data structure [9].	GFS supports a hierarchical directories data structure and access by path names [9].
Types of Nodes used	NameNode and DataNode	Chunk-server and MasterNode

Hardware used	Commodity Hardware or Server	Commodity Hardware or Server
Append Opration	Only supports append operation	supports append operation and we can also append base on offset.
Database Files	Hbase	Bigtable is the database
Delete Opration and Garbage Collection	First, deleted files are renamed and store in particular folder then finally remove using garbage collection method.	GFS has unique garbage collection method in which we cannot reclaiminstantly. It will rename the namespace It will delete after the 3 days during the second scaned.
Default size	HDFS has by default DataNode size 128 MB but it can be change by the user	GFS has by default chunk size 64 MB but it can be change by the user
Snapshots	HDFS allowed upto 65536 snapshots for each directory in HDFS 2.	In GFS Each directories and files can be snapshotted.
Meta-Data	Meta-Data information managed by NameNode.	Meta-Data information managed by MasterNode.
Data Integrity	Data Integrity maintain in between NameNode and DataNode.	Data Integrity maintain in between MasterNode and Chunk-Server.
Replication	There are two time replicas created by default in GFS [10].	There are three time replicas created by default in GFS [10].
Communication	Pipelining is used to data transfer over the TCP protocol.	RPC based protocol used on top of TCP\IP.
Cache management	HDFS provide the distributed cache facility using Mapreduse framework	GFS does not provide the cache facility

Note: Main objectives of GFS is to handle chunk or file name space, chunk server.