Prathamesh Arvind Jadhav

Cloud Technologies and Advancements

Hadoop-HDFS in Cloud Computing

□ What is Hadoop?

Apache Hadoop is an open-source framework designed to **store and process large datasets** in a **distributed computing environment**. It is scalable, fault-tolerant, and efficient for big data processing.

□ What is HDFS (Hadoop Distributed File System)?

HDFS is the **primary storage system** of Hadoop. It allows data to be stored across multiple machines and enables **parallel processing** using **MapReduce** or other processing frameworks.

□ Key Features of HDFS

Feature	Description
Distributed Storage	Data is split into blocks and stored across multiple nodes.
Fault Tolerance	Automatically replicates data blocks to handle node failures.
High Throughput	Optimized for high data access speeds using large block sizes.
Scalability	New nodes can be easily added without downtime.
Write-once-read-many	Ideal for batch processing as data is written once and read many times.

Prathamesh Arvind Jadhav

□ HDFS Architecture

1. **NameNode (Master)**:
   - Manages file system namespace.
   - Keeps metadata (location of blocks, permissions).
   - **Single point of coordination** (critical component).
2. **DataNode (Worker)**:
   - Stores actual data blocks.
   - Sends heartbeat and block reports to NameNode.
   - Reads/writes blocks as requested.
3. **Secondary NameNode**:
   - Periodically pulls metadata and creates checkpoints.
   - **Not a backup** NameNode (common misconception).

□ How HDFS Works in Cloud Computing

□ Integration of HDFS with Cloud:

1. **Elastic Scalability**:
   - Cloud platforms (AWS, Azure, GCP) allow dynamic scaling of HDFS nodes.
2. **Cost-Effective Storage**:
   - Use of cloud object stores (like Amazon S3) to complement or replace HDFS.
3. **High Availability**:
   - Cloud infrastructure supports HDFS high availability (HA) setups.
4. **On-demand Resources**:
   - Easily provision Hadoop clusters using cloud services like Amazon EMR, Google Dataproc.

□ HDFS Workflow Example

1. A large file (e.g., 1 GB) is split into **blocks** (e.g., 128 MB).
2. Each block is stored in **multiple DataNodes** (default replication factor = 3).
3. NameNode manages the metadata (not the actual data).
4. When reading, the client contacts the NameNode for block locations and directly reads from DataNodes.

□ HDFS vs Cloud Storage

Feature	HDFS	Cloud Storage (e.g., S3, Azure Blob)
Performance	High throughput for big data	Good for object storage, not compute-intensive
Scalability	Scalable but manual	Auto-scalable
Management	Requires admin expertise	Managed by cloud provider
Use Case	Hadoop jobs, big data	Backup, web apps, serverless, ML pipelines

---

□ Applications of Hadoop-HDFS in Cloud

- **Big Data Analytics**
- **Machine Learning pipelines**
- **Log Processing**
- **ETL Workflows**
- **IoT Data Storage and Analysis**

□ Summary

- **Hadoop HDFS** is a foundational storage system for big data processing.
- In **cloud computing**, HDFS is either deployed on cloud VMs or replaced/augmented with cloud-native storage solutions.
- Combines **high performance** of Hadoop with the **flexibility and scalability** of cloud.

MapReduce

□ What is MapReduce?

MapReduce is a **programming model** and **processing technique** for **large-scale data processing**. It was developed by Google and popularized through **Hadoop**. The model is designed to **process massive amounts of data** in a **parallel, distributed manner** across a Hadoop cluster.

□ Basic Concept

The MapReduce process is divided into two main phases:

1. **Map Phase**
2. **Reduce Phase**

Each phase has a **key-value pair** mechanism:

Phase	Input	Output
Map	(Key1, Value1)	→ (Key2, Value2)

Phase	Input	Output
Reduce	(Key2, List<Value2>)	→ (Key3, Value3)

□ Steps of MapReduce in Detail

□ 1. Input Splitting

- The input data is split into **blocks** (typically 128MB or 64MB).
- Each block is processed independently in parallel by a **Map task**.

□ 2. Map Phase

- Takes a block of input and processes it into key-value pairs.
- Example: For a word count application, it emits (word, 1) for each word.

□ 3. Shuffling and Sorting

- After the Map phase, all emitted (key, value) pairs are grouped by key.
- The system **shuffles** the data to ensure all values for the same key are sent to the same reducer.
- Keys are **sorted** before reaching the Reduce phase.

□ 4. Reduce Phase

- Aggregates values for each key.
- Example: Adds up all counts for each word.

□ 5. Output Phase

- The final output is stored in **HDFS** or any storage system.
- Output format is also key-value pairs (e.g., word -> count).

□ MapReduce Job Components

Component	Role
JobTracker (deprecated in newer versions)	Manages resources and schedules tasks (YARN replaces this).
TaskTracker (deprecated)	Executes individual tasks (now handled by NodeManager).
YARN (Yet Another Resource Negotiator)	Modern resource manager for Hadoop jobs.
InputFormat	Defines input source and splitting mechanism.
OutputFormat	Defines how final output is written.

□ Why Use MapReduce?

- **Scalable**: Handles petabytes of data.
- **Fault-Tolerant**: Failed tasks are automatically retried.
- **Parallel Processing**: Efficient utilization of resources.
- **Simplified Programming**: Developers just focus on Map and Reduce logic.

□ Limitations of MapReduce

- **Not real-time**: Batch-oriented, high latency.
- **Hard to debug** and monitor.
- **I/O intensive**: Shuffling and disk writing can be costly.
- **Limited flexibility** for complex workflows (replaced in many cases by Spark).

□ MapReduce vs Spark

Feature	MapReduce	Apache Spark
Processing	Disk-based (Batch)	In-memory (Faster)
Speed	Slower due to disk I/O	Up to 100x faster
Ease of Use	Java-heavy, verbose	APIs in Python, Scala, Java
Use Case	Legacy big data jobs	Real-time + batch

□ Applications of MapReduce

- Log analysis
- Data mining
- Web indexing (Google used it initially)
- ETL (Extract, Transform, Load) jobs
- Sentiment analysis
- Fraud detection

□ Summary

- **MapReduce** enables **scalable and parallel data processing** using a simple key-value model.
- It breaks tasks into Map → Shuffle → Reduce steps.
- Though somewhat outdated, it's foundational in **big data ecosystems** and is often replaced by modern tools like **Apache Spark** for better performance and flexibility.

□ Overview of VirtualBox

Oracle VM VirtualBox is an open-source, cross-platform virtualization software that allows users to run multiple operating systems simultaneously on a single physical machine. It acts like a **hypervisor**, enabling the creation and management of **virtual machines (VMs)**.

□ What is Virtualization?

Virtualization is the process of creating a **virtual version** of something, such as hardware platforms, storage devices, or network resources. VirtualBox enables **hardware-level virtualization**, allowing an operating system (guest) to run within another (host) as if it were installed directly on the hardware.

□ Key Components of VirtualBox

1. Host Operating System

- The OS installed on the physical machine.

- Examples: Windows, macOS, Linux, Solaris.

2. Guest Operating System

- The OS that runs inside VirtualBox as a virtual machine.
- Can be different from the host OS (e.g., Ubuntu guest on a Windows host).

3. Hypervisor Layer

- VirtualBox acts as a **Type 2 hypervisor** (hosted).
- It runs on top of the host OS and uses host resources to create and manage VMs.

4. Virtual Machine (VM)

- A software-based emulation of a computer.
- Contains virtual hardware like CPU, RAM, disk, display, network card, etc.

□ Features of VirtualBox

□ Cross-platform support

- Runs on Windows, macOS, Linux, and Solaris hosts.
- Supports various guest OS types (Windows, Linux, BSD, macOS unofficially).

□ Snapshots

- Allows saving the current state of a VM.
- You can roll back to a snapshot in case of issues.

□ Shared Folders and Clipboard

- Facilitates file and clipboard sharing between host and guest.

□ Guest Additions

- A set of drivers and applications installed in the guest OS.
- Provides better integration, e.g., auto-resize, drag-and-drop, seamless mode.

□ USB Device Support

- VM can access USB devices connected to the host.

□ Networking Modes

- NAT (default), Bridged, Internal, Host-only, etc.
- Allows VMs to connect to the internet or each other.

□ Virtual Hard Disks

- Formats like VDI, VMDK, VHD supported.
- Can use dynamic or fixed-size storage.

□□ Common Use Cases

1. **Software Development & Testing**
    - Run multiple environments for compatibility testing.
2. **Learning & Experimentation**
    - Practice Linux or other OS without installing it natively.
3. **Server Virtualization**
    - Host web servers, database servers, etc., for small-scale use.
4. **Isolated Sandbox Environments**
    - Test suspicious software or conduct malware analysis.

□ VirtualBox vs Other Virtualization Tools

Feature	VirtualBox	VMware Workstation	Hyper-V

Feature	VirtualBox	VMware Workstation	Hyper-V
License	Open-source	Proprietary	Built-in (Windows Pro)
Host OS Compatibility	Windows, Linux, macOS	Windows, Linux	Windows only
Guest OS Support	Broad	Broad	Mostly Windows
Performance	Moderate	Higher (paid)	High

□ Installing & Using VirtualBox

Step-by-Step:

1. Download and install VirtualBox from https://www.virtualbox.org
2. Create a new VM:
    - Choose OS type and version
    - Allocate RAM and CPU
    - Create or choose virtual hard disk
3. Install guest OS from ISO image
4. Install **Guest Additions** inside the VM for full functionality
5. Use snapshot and network features as needed

□ Limitations

- Slightly lower performance compared to native OS
- macOS guest is not officially supported on non-Apple hardware

- Limited GPU passthrough support (not ideal for heavy 3D/GPU use)

 Summary

Aspect	Description
Type	Type 2 Hypervisor
Main Use	Run multiple OSes on one machine
Key Benefit	Free, open-source, cross-platform
Ideal For	Developers, testers, learners
Downsides	Slight performance drop, limited macOS guest support

Google Apps

 What is Google Apps (Google Workspace)?

Google Apps, now officially called **Google Workspace**, is a suite of cloud-based applications and services designed to help individuals, businesses, educators, and organizations **collaborate, communicate, and manage data and workflows efficiently**—all through the web.

 Core Components of Google Workspace

1. Gmail

- Email service with powerful spam filtering and smart categorization.
- Custom domain support (e.g., yourname@yourcompany.com).
- Integrated with Google Chat and Meet.

2. Google Drive

- Cloud storage system for saving and sharing files.
- Offers **15 GB free**, with paid plans for more.
- Supports multiple file types (PDF, DOCX, PPT, ZIP, etc.).
- Real-time collaboration and version history.

3. Google Docs, Sheets, and Slides

- Web-based alternatives to Microsoft Word, Excel, and PowerPoint.
- Enable real-time co-editing and commenting.
- Auto-save feature and seamless file sharing.
- Offline access supported.

4. Google Meet

- Video conferencing tool integrated with Gmail and Calendar.
- Supports screen sharing, captions, breakout rooms.
- Secure, with encryption and admin controls.

5. Google Calendar

- Time management and scheduling tool.
- Shareable calendars and automatic event invitations.
- Syncs across devices.

6. Google Forms

- Tool for creating surveys, quizzes, and feedback forms.
- Auto-populates responses into Google Sheets.
- Useful for event registrations, assessments, etc.

7. Google Sites

- Simple website builder for internal portals, project pages, etc.
- No coding required.
- Drag-and-drop design.

8. Google Chat

- Instant messaging tool integrated with Gmail.
- Organize messages by rooms (groups) or threads.
- Supports files, bots, and task lists.

9. Google Keep

- Note-taking tool.
- Syncs across devices.
- Supports images, lists, and voice memos.

10. Google Admin Console (for organizations)

- Central hub for managing users, devices, apps, and permissions.
- Security settings, 2FA enforcement, and audit logs.

□ Security and Privacy Features

- Two-Factor Authentication (2FA)
- Endpoint Management (for lost/stolen devices)
- Data Loss Prevention (DLP)
- Role-based Access Control
- Built-in spam, phishing, and malware protection

□ Google Workspace Plans (as of 2024)

Plan	Features	Ideal For
Free	Gmail, Drive (15 GB), Docs, Sheets, etc.	Individuals
Business Starter	Custom email, 30 GB storage/user, Meet (100 participants)	Small businesses
Business Standard	2 TB/user, Meet (150 users + recording), enhanced collaboration	Mid-sized organizations
Business Plus	5 TB/user, advanced security & audit logs	Larger organizations
Enterprise	Custom solutions, unlimited storage, enterprise-grade security	Enterprises

---

□ Benefits of Google Apps

- □ **Accessibility**: Use on any device with a browser or app.
- □ **Collaboration**: Multiple users can edit the same document in real time.
- □ **Scalability**: Suitable for individuals, startups, and large enterprises.
- □ **Integration**: Deeply integrated tools that communicate with each other.
- □ **Uptime & Reliability**: Google guarantees **99.9% uptime**.
- □ **Cost-effective**: Subscription-based pricing, no on-premises hardware.

□ Common Use Cases

Use Case	Tools Used
Team Collaboration	Docs, Sheets, Meet, Drive, Chat
Project Management	Calendar, Tasks, Chat, Keep
Education	Classroom (Google App), Forms, Meet
Marketing Campaigns	Slides, Gmail, Sites, Analytics
Surveys and Feedback	Google Forms + Sheets
Website and Portal Creation	Google Sites

□ Google Apps vs. Microsoft 365

Feature	Google Workspace	Microsoft 365
Editing	Real-time cloud-based	Desktop + cloud (hybrid)
Email	Gmail	Outlook
Storage	Google Drive	OneDrive
Collaboration	Excellent real-time features	Good, but not as seamless
Interface	Clean and modern	More traditional (MS-style)

□ Summary

Parameter	Details

Parameter	Details
Developer	Google Inc.
Type	Cloud-based productivity suite
Former Name	Google Apps, G Suite
Key Strength	Real-time collaboration and integration
Target Audience	Individuals, SMBs, Enterprises, Schools

□ What is OpenStack?

OpenStack is an **open-source cloud computing platform** used to build and manage **public and private clouds**. It provides **Infrastructure as a Service (IaaS)** by pooling compute, storage, and networking resources into a unified platform that users can manage through a **dashboard**, **CLI**, or **RESTful APIs**.

- Originally developed by **NASA and Rackspace in 2010**
- Now maintained by the **Open Infrastructure Foundation**
- Used by organizations like Walmart, CERN, PayPal, and more

□ Key Purpose of OpenStack

- Create and manage large pools of **virtual machines, containers, storage, and networks**
- Enable **self-service** provisioning for developers and admins
- Provide a scalable, flexible, and **vendor-neutral** cloud environment

---

□ OpenStack Architecture Overview

OpenStack follows a **modular architecture**, where each module (project) handles a specific cloud function. These components interact using **REST APIs** and message queues (usually **RabbitMQ**).

□ Core Components of OpenStack

Component	Role
Nova	Compute management – handles VM lifecycle
Neutron	Networking – manages IPs, networks, subnets, routing
Cinder	Block Storage – provides persistent volumes for VMs
Glance	Image Service – stores and retrieves VM images
Keystone	Identity service – handles authentication and authorization
Horizon	Web dashboard – graphical interface for OpenStack services
Swift	Object Storage – scalable storage for unstructured data
Heat	Orchestration – automates cloud app deployment

Component	Role
	using templates
Ceilometer/Gnocchi	Telemetry – monitors and meters resource usage
Barbican	Key management – manages secrets and certificates
Ironic	Bare-metal provisioning

□ How OpenStack Works – High-Level Flow

1. **User logs in** via Horizon dashboard or API, authenticated by **Keystone**.
2. **Nova** schedules and spins up a **VM** using an image from **Glance** and attaches a block volume from **Cinder**.
3. The VM is connected to a virtual network via **Neutron**.
4. The VM can read/write large data to **Swift** (object storage).
5. All actions are monitored via **Ceilometer/Gnocchi** for billing/monitoring.
6. Orchestration templates from **Heat** can automate deployment of multi-tier apps.

□ Typical Deployment

OpenStack can be deployed across multiple physical nodes:

- **Controller Node**: Runs services like Keystone, Glance, Horizon, Neutron-server
- **Compute Node(s)**: Runs hypervisor (e.g., KVM) and Nova Compute service

- **Storage Node(s)**: Provides block (Cinder) or object (Swift) storage
- **Network Node**: Handles routing, NAT, DHCP, etc.

□ Key Features of OpenStack

- **Modular & Extensible**: Add/remove components as needed
- **Vendor-Neutral**: Supports multiple hypervisors, hardware, and plug-ins
- **Self-Service**: Users can launch and manage their own instances
- **API-Driven**: Integrates with DevOps tools like Terraform, Ansible
- **Community Supported**: Thousands of developers contribute globally

---

□ Use Cases

1. **Private Cloud Infrastructure** – Enterprises running internal workloads
2. **Public Cloud Providers** – e.g., OVH, DreamHost
3. **Hybrid Clouds** – Integrating with AWS, Azure
4. **Telco NFV** – Used in network function virtualization by telecoms
5. **Research & HPC** – Used by universities and labs

□ Security in OpenStack

- **Keystone** supports role-based access control (RBAC)
- Projects like **Barbican** manage encryption keys and secrets
- TLS, SSL, and token-based auth supported
- Isolated tenants and networks for user-level security

□ Summary Table

Feature	Description
Type	IaaS (Infrastructure as a Service)
Developed By	Open Infrastructure Foundation
Components	Nova, Neutron, Glance, Cinder, Keystone, etc.
Interfaces	CLI, REST API, Horizon GUI
Hypervisors	KVM, Xen, Hyper-V, VMware
Storage Types	Block (Cinder), Object (Swift)
Deployment Tools	DevStack, PackStack, TripleO, Juju, Ansible
Integration	Terraform, Kubernetes, Docker, Ceph

□ Federation in the Cloud – Overview

□ What is Cloud Federation?

Cloud Federation refers to a **collaboration between multiple cloud service providers (CSPs)** or **cloud environments** (public/private/hybrid) that allows **resource sharing**, **interoperability**, **unified access**, and **seamless user experience** across different cloud platforms.

It enables **users or organizations to access and use services from multiple clouds** as if they were part of a single system.

Prathamesh Arvind Jadhav

□ Why is Cloud Federation Important?

- Avoid vendor lock-in
- Balance loads and optimize resource usage
- Ensure service continuity and fault tolerance
- Enable multi-cloud and hybrid cloud strategies
- Centralized identity and access management across clouds

□ Key Elements of Cloud Federation

1. **Federated Identity Management (FIM)**
   - Enables **single sign-on (SSO)** across clouds
   - Users authenticate once to access resources across multiple platforms
2. **Federated Resource Management**
   - Share VMs, storage, and networks between cloud providers
3. **Federated Networking**
   - Connect private/public clouds under unified networking
4. **Service-Level Agreement (SLA) Federation**
   - Ensures consistent QoS and policies across clouds

□ Types of Federation in Cloud

1. Identity Federation (Authentication/Authorization Federation)

- Allows users to use a **single identity** across multiple clouds.
- Managed through protocols like:
  - **SAML** (Security Assertion Markup Language)
  - **OAuth2 / OpenID Connect**
  - **LDAP integration**

Example: Logging into AWS using Google or Azure credentials via SAML.

---

2. Data Federation

- Enables **data integration and access** from **multiple cloud data sources** as if it were a **single data source**.
- Common in **cloud data warehouses** and **analytics platforms**.

Example: A BI tool accessing both AWS Redshift and Google BigQuery in a unified view.

---

3. Infrastructure Federation

- Combines **compute/storage/network resources** across multiple clouds.
- Enables **VM migration**, **load balancing**, and **failover** between cloud infrastructures.

Example: An enterprise uses both AWS and Azure to host services and can move workloads between them.

---

4. Service Federation

- Integrates **services or applications** across clouds.
- Useful in **multi-cloud microservices** or **API-based architectures**.

Example: A front-end on AWS calling a backend API hosted on GCP.

---

5. Cloud Provider Federation

- Cloud providers partner to offer a **combined or unified cloud service**.
- Users gain **access to a shared pool of services** across federated clouds.

Example: OpenStack-based clouds forming a **federated cloud consortium**.

□ Technologies and Standards Used

Area	Standards/Tools
Identity	SAML, OAuth2, OpenID Connect, LDAP
Resource Sharing	REST APIs, Terraform, OpenStack
Data Federation	Presto, Apache Drill, BigQuery Federation
Network Federation	VPN, VPC peering, SD-WAN
Orchestration	Kubernetes, Federation v2, Heat

□ Benefits of Cloud Federation

- □ Seamless cross-cloud experience
- □ Reduced vendor dependency
- □ Better fault tolerance and disaster recovery
- □ Optimal cost/resource management
- □ Unified governance and compliance

□ Challenges of Cloud Federation

- □ Complexity in setup and integration

- □ Differences in APIs and service models
- □ Security risks in federated identity
- □ SLA mismatches between providers
- □ Compliance issues (e.g., GDPR, HIPAA)

□ Real-World Examples

Example	Description
EduGAIN	Federation of academic institutions for shared cloud access
OpenStack Federation	Linking multiple OpenStack clouds for shared identity and workloads
Google Workspace + Azure AD	Use Azure credentials to log in to Google services
Hybrid Cloud Solutions (e.g., VMware Cloud)	Integrates on-premise and cloud infrastructures

---

□ Summary

Aspect	Details
Definition	Collaboration of multiple clouds to share services/resources
Key Feature	Unified identity, resource sharing, SLA consistency
Major Types	Identity, Data, Infrastructure, Service, Provider Federation

Aspect	Details
Benefits	Flexibility, redundancy, cost optimization, scalability
Tools/Protocols	SAML, OAuth2, REST, Kubernetes, OpenStack