JEPPIAAR INSTITUTE OF TECHNOLOGY

"Self Belief | Self Discipline | Self Respect"

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

LECTURE NOTES

CS8791 / CLOUD COMPUTING

(2017 Regulation)

Year/Semester: IV / VII

Prepared by

Dr. K. Tamilarasi, Professor / Dept. of CSE.

UNIT V CLOUD TECHNOLOGIES AND ADVANCEMENTS

Hadoop – MapReduce – Virtual Box -- Google App Engine – Programming Environment for Google App Engine – OpenStack – Federation in the Cloud – Four Levels of Federation – Federated Services and Applications – Future of Federation.

5.1 Hadoop

- Hadoop is an open source implementation of MapReduce coded and released in Java (rather than C) by Apache.

- The Hadoop implementation of MapReduce uses the Hadoop Distributed File System (HDFS) as its underlying layer rather than GFS.

- The Hadoop core is divided into two fundamental layers:

    - MapReduce engine
    - HDFS

- The MapReduce engine is the computation engine running on top of HDFS as its data storage manager.

- HDFS is a distributed file system inspired by GFS that organizes files and stores their data on a distributed computing system.

- HDFS Architecture: HDFS has a master/slave architecture containing a single NameNode as the master and a number of DataNodes as workers (slaves).

- To store a file in this architecture, HDFS splits the file into fixed-size blocks (e.g., 64 MB) and stores them on workers (DataNodes).

- The mapping of blocks to DataNodes is determined by the NameNode.

- The NameNode (master) also manages the file system's metadata and namespace.

- In such systems, the namespace is the area maintaining the metadata and metadata refers to all the information stored by a file system that is needed for overall management of all files.

- For example, NameNode in the metadata stores all information regarding the location of input splits/blocks in all DataNodes.

- Each DataNode, usually one per node in a cluster, manages the storage attached to the node. Each DataNode is responsible for storing and retrieving its file blocks.

- HDFS Features: Distributed file systems have special requirements, such as performance, scalability, concurrency control, fault tolerance and security requirements, to operate efficiently.

- However, because HDFS is not a general purpose file system, as it only executes specific types of applications, it does not need all the requirements of a general distributed file system.

- One of the main aspects of HDFS is its fault tolerance characteristic. Since Hadoop is designed to be deployed on low-cost hardware by default, a hardware failure in this system is considered to be common rather than an exception.

- Hadoop considers the following issues to fulfill reliability requirements of the file system

    - Block replication: To reliably store data in HDFS, file blocks are replicated in this system. The replication factor is set by the user and is three by default.

- Replica placement: The placement of replicas is another factor to fulfill the desired fault tolerance in HDFS.
- Heartbeat and Block report messages: Heartbeats and Block reports are periodic messages sent to the NameNode by each DataNode in a cluster.
- Applications run on HDFS typically have large data sets, individual files are broken into large blocks (e.g., 64 MB) to allow HDFS to decrease the amount of metadata storage required per file.
- This provides two advantages:

    - The list of blocks per file will shrink as the size of individual blocks increases.
    - Keeping large amounts of data sequentially within a block provides fast streaming reads of data.

- HDFS Operation: The control flow of HDFS operations such as write and read can properly highlight roles of the NameNode and DataNodes in the managing operations

    - To read a file in HDFS, a user sends an “open” request to the NameNode to get the location of file blocks.
    - For each file block, the NameNode returns the address of a set of DataNodes containing replica information for the requested file.
    - The number of addresses depends on the number of block replicas. Upon receiving such information, the user calls the read function to connect to the closest DataNode containing the first block of the file.
    - After the first block is streamed from the respective DataNode to the user, the established connection is terminated and the same process is repeated for all blocks of the requested file until the whole file is streamed to the user.
    - To write a file in HDFS, a user sends a “create” request to the NameNode to create a new file in the file system namespace.
    - If the file does not exist, the NameNode notifies the user and allows him to start writing data to the file by calling the write function.
    - The first block of the file is written to an internal queue termed the data queue while a data streamer monitors its writing into a DataNode.

- Since each file block needs to be replicated by a predefined factor, the data streamer first sends a request to the NameNode to get a list of suitable DataNodes to store replicas of the first block.
- The steamer then stores the block in the first allocated DataNode.
- Afterward, the block is forwarded to the second DataNode by the first DataNode.
- The process continues until all allocated DataNodes receive a replica of the first block from the previous DataNode.
- Once this replication process is finalized, the same process starts for the second block and continues until all blocks of the file are stored and replicated on the file system.

5.2 MapReduce

- The topmost layer of Hadoop is the MapReduce engine that manages the data flow and control flow of MapReduce jobs over distributed computing systems.

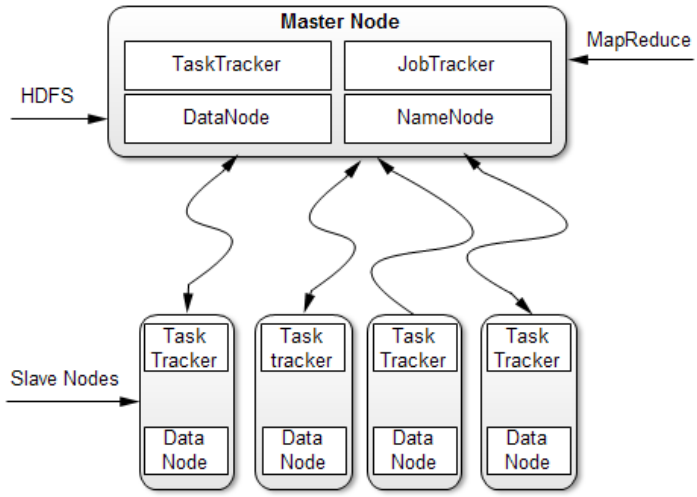


Figure 5.1 HDFS and MapReduce Architecture

- Figure 5.1 shows the MapReduce engine architecture cooperating with HDFS.

- Similar to HDFS, the MapReduce engine also has a master/slave architecture consisting of a single JobTracker as the master and a number of TaskTrackers as the slaves (workers).

- The JobTracker manages the MapReduce job over a cluster and is responsible for monitoring jobs and assigning tasks to TaskTrackers.

- The TaskTracker manages the execution of the map and/or reduce tasks on a single computation node in the cluster.

- Each TaskTracker node has a number of simultaneous execution slots, each executing either a map or a reduce task.

- Slots are defined as the number of simultaneous threads supported by CPUs of the TaskTracker node.

- For example, a TaskTracker node with N CPUs, each supporting M threads, has M * N simultaneous execution slots.

- It is worth noting that each data block is processed by one map task running on a single slot.

- Therefore, there is a one to one correspondence between map tasks in a TaskTracker and data blocks in the respective DataNode.

Running a Job in Hadoop

- Three components contribute in running a job in this system:

    - User node
    - JobTracker
    - TaskTrackers

- The data flow starts by calling the runJob (conf) function inside a user program running on the user node, in which conf is an object containing some tuning parameters for the MapReduce framework and HDFS.

- The runJob (conf) function and conf are comparable to the MapReduce (Spec, &Results) function and Spec in the first implementation of MapReduce by Google.

- Figure 5.2 depicts the data flow of running a MapReduce job in Hadoop.

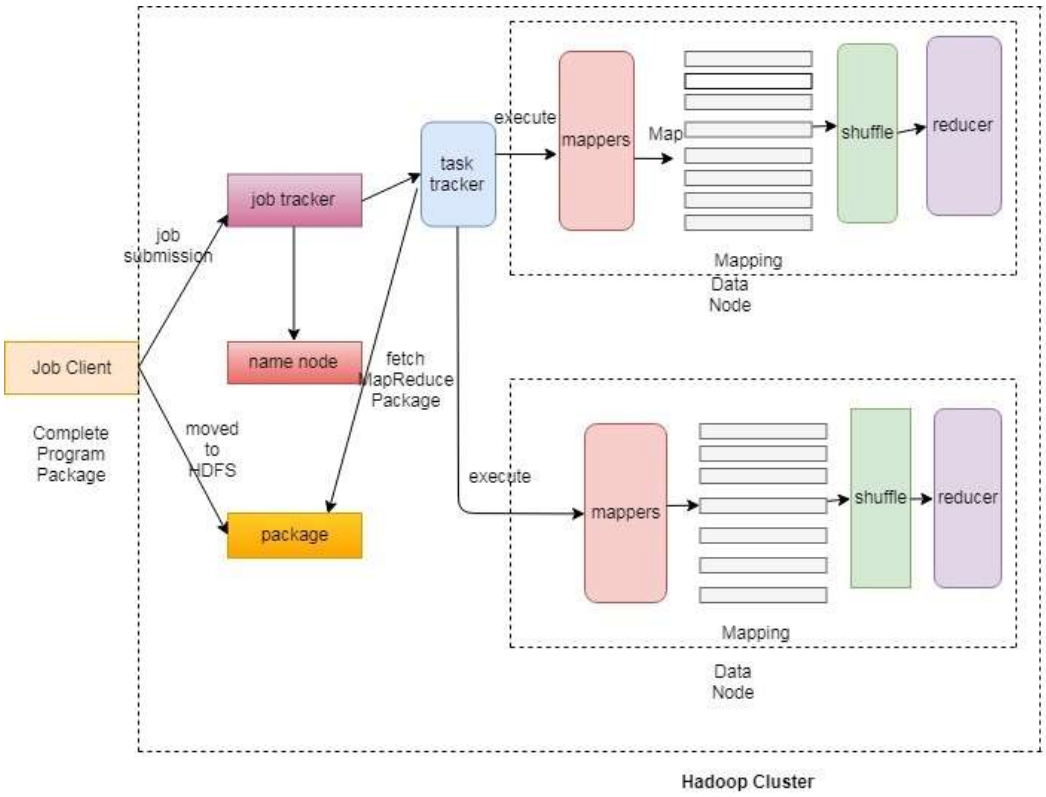


Figure 5.2 Data flow in Hadoop

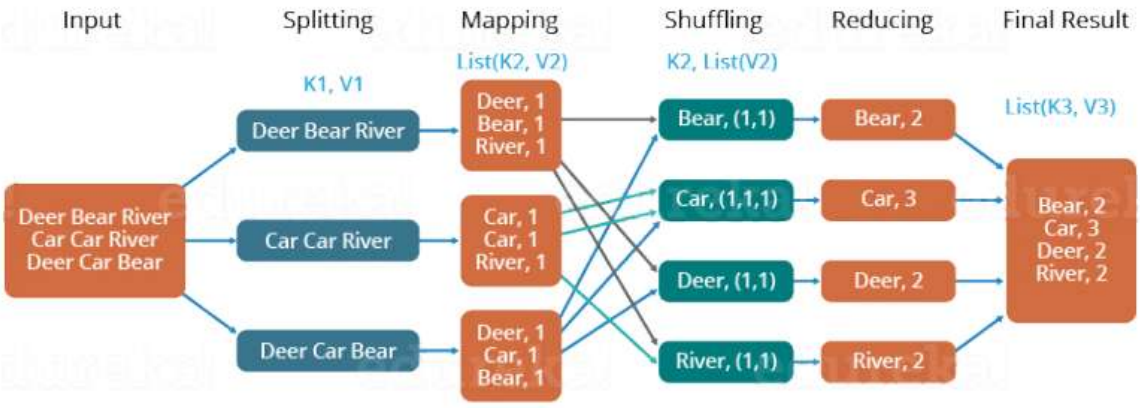
- Job Submission Each job is submitted from a user node to the JobTracker node that might be situated in a different node within the cluster through the following procedure:

    - A user node asks for a new job ID from the JobTracker and computes input file splits.

- The user node copies some resources, such as the job's JAR file, configuration file, and computed input splits, to the JobTracker's file system.
- The user node submits the job to the JobTracker by calling the submitJob() function.
- Task assignment The JobTracker creates one map task for each computed input split by the user node and assigns the map tasks to the execution slots of the TaskTrackers.
  - The JobTracker considers the localization of the data when assigning the map tasks to the TaskTrackers.
  - The JobTracker also creates reduce tasks and assigns them to the TaskTrackers.
  - The number of reduce tasks is predetermined by the user, and there is no locality consideration in assigning them.

- Task execution The control flow to execute a task (either map or reduce) starts inside the TaskTracker by copying the job JAR file to its file system.
- Instructions inside the job JAR file are executed after launching a Java Virtual Machine (JVM) to run its map or reduce task.
- Task running check A task running check is performed by receiving periodic heartbeat messages to the JobTracker from the TaskTrackers.
- Each heartbeat notifies the JobTracker that the sending TaskTracker is alive, and whether the sending TaskTracker is ready to run a new task.

Example:



- First, we divide the input into three splits as shown in the figure. This will distribute the work among all the map nodes.
- Then, we tokenize the words in each of the mappers and give a hardcoded value (1) to each of the tokens or words. The rationale behind giving a hardcoded value equal to 1 is that every word, in itself, will occur once.
- Now, a list of key-value pair will be created where the key is nothing but the individual words and value is one. So, for the first line (Dear Bear River) we have 3 key-value pairs – Dear, 1; Bear, 1; River, 1. The mapping process remains the same on all the nodes.
- After the mapper phase, a partition process takes place where sorting and shuffling happen so that all the tuples with the same key are sent to the corresponding reducer.
- So, after the sorting and shuffling phase, each reducer will have a unique key and a list of values corresponding to that very key. For example, Bear, [1,1]; Car, [1,1,1].., etc.
- Now, each Reducer counts the values which are present in that list of values. As shown in the figure, reducer gets a list of values which is [1,1] for the key Bear. Then, it counts the number of ones in the very list and gives the final output as – Bear, 2.
- Finally, all the output key/value pairs are then collected and written in the output file.

5.3 Virtual Box

- Oracle VM VirtualBox is a cross platform virtualization application.

- For one thing, it installs on the existing Intel or AMD-based computers, whether they are running Windows, Mac OS X, Linux, or Oracle Solaris operating systems (OSes).

- Secondly, it extends the capabilities of existing computer so that it can run multiple OSes, inside multiple virtual machines, at the same time.

- As an example, the end user can run Windows and Linux on your Mac, run Windows Server 2016 on your Linux server, run Linux on your Windows PC, and so on, all alongside the existing applications.

- The user can install and run as many virtual machines.

- The only practical limits are disk space and memory.

- Oracle VM VirtualBox is deceptively simple yet also very powerful.

- It can run everywhere from small embedded systems or desktop class machines all the way up to datacenter deployments and even Cloud environments.

- Virtual Box is created by Innotek and it was acquired by Sun Microsystems. In 2010, Virtual Box was acquired by Oracle.

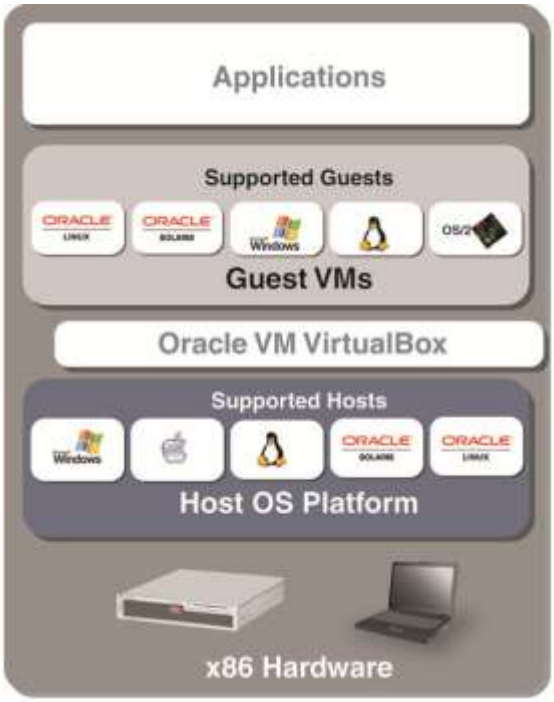


Figure 5.3 architecture of Virtual Box

- Virtual Box supported in Windows, macOS. Linux, Solaris and Open Solaris.

- Figure 5.3 depicts the architecture of Virtual Box

- The user can independently configure each VM and run it under a choice of software-based virtualization or hardware assisted virtualization if the underlying host hardware supports this.

- The host OS and guest OSs and applications can communicate with each other through a number of mechanisms including a common clipboard and a virtualized network facility.

- Guest VMs can also directly communicate with each other if configured to do so.

- The software based virtualization was dropped starting with VirtualBox 6.1. In earlier versions the absence of hardware assisted virtualization, VirtualBox adopts a standard software-based virtualization approach.

- This mode supports 32 bit guest OSs which run in rings 0 and 3 of the Intel ring architecture.

    - The system reconfigures the guest OS code, which would normally run in ring 0, to execute in ring 1 on the host hardware.
    - Because this code contains many privileged instructions which cannot run natively in ring 1, VirtualBox employs a Code Scanning and Analysis Manager (CSAM) to scan the ring 0 code recursively before its first execution to identify problematic instructions and then calls the Patch Manager (PATM) to perform in-situ patching.
    - This replaces the instruction with a jump to a VM-safe equivalent compiled code fragment in hypervisor memory.

    - The guest user mode code, running in ring 3, generally runs directly on the host hardware in ring 3.

- In both cases, VirtualBox uses CSAM and PATM to inspect and patch the offending instructions whenever a fault occurs.

- VirtualBox also contains a dynamic recompiler, based on QEMU to recompile any real mode or protected mode code entirely.

- Hardware assisted virtualization is starting with version 6.1, VirtualBox only supports.

- VirtualBox supports both Intel VT-X and AMD-V hardware assisted virtualization.

- Making use of these facilities, VirtualBox can run each guest VM in its own separate address-space.

- The guest OS ring 0 code runs on the host at ring 0 in VMX non-root mode rather than in ring 1.

- Until then, VirtualBox specifically supported some guests (including 64 bit guests, SMP guests and certain proprietary OSs) only on hosts with hardware-assisted virtualization

- The system emulates hard disks in one of three disk image formats:

    - VDI: This format is the VirtualBox-specific VirtualBox Disk Image and stores data in files bearing a ".vdi" .
    - VMDK: This open format is used by VMware products and stores data in one or more files bearing ".vmdk" filename extensions. A single virtual hard disk may span several files.
    - VHD: This format is used by Windows Virtual PC and Hyper-V and it is the native virtual disk format of the Microsoft Windows operating system. Data in this format are stored in a single file bearing the ".vhd" filename extension.

- A VirtualBox virtual machine can, therefore, use disks previously created in VMware or Microsoft Virtual PC, as well as its own native format.

- VirtualBox can also connect to iSCSI targets and to raw partitions on the host, using either as virtual hard disks.

- VirtualBox has supported Open Virtualization Format (OVF).

- By default, VirtualBox provides graphics support through a custom virtual graphics-card

- For an Ethernet network adapter, VirtualBox virtualizes these Network Interface Cards.

    - AMD PCnet PCI II
    - AMD PCnet-Fast III
    - Intel Pro/1000 MT Desktop
    - Intel Pro/1000 MT Server
    - Intel Pro/1000 T Server
    - Paravirtualized network adapter

- For a sound card, VirtualBox virtualizes Intel HD Audio.

- A USB controller is emulated so that any USB devices attached to the host can be seen in the guest.

- Oracle VM VirtualBox was designed to be modular and flexible.

- When the Oracle VM VirtualBox graphical user interface (GUI) is opened and a VM is started, at least the following three processes are running:

    - VBoxSVC is Oracle VM VirtualBox service process which always runs in the background. This process is started automatically by the first Oracle VM VirtualBox client process and exits a short time after the last client exits.

    - The first Oracle VM VirtualBox service can be the GUI, VBoxManage, VBoxHeadless, the web service amongst others.

    - The service is responsible for bookkeeping, maintaining the state of all VMs, and for providing communication between Oracle VM VirtualBox components.

- Oracle VM VirtualBox comes with comprehensive support for third-party developers.

- The Main API of Oracle VM VirtualBox exposes the entire feature set of the virtualization engine.

- The Main API is made available to C++ clients through COM on Windows hosts or XPCOM on other hosts. Bridges also exist for SOAP, Java and Python.

5.4 Google App Engine

- Google has the world's largest search engine facilities.

- The company has extensive experience in massive data processing that has led to new insights into data-center design and novel programming models that scale to incredible sizes.

- Google platform is based on its search engine expertise.

- Google has hundreds of data centers and has installed more than 460,000 servers worldwide.

- For example, 200 Google data centers are used at one time for a number of cloud applications.

- Data items are stored in text, images, and video and are replicated to tolerate faults or failures.

- Google's App Engine (GAE) which offers a PaaS platform supporting various cloud and web applications.

- Google has pioneered cloud development by leveraging the large number of data centers it operates.

- For example, Google pioneered cloud services in Gmail, Google Docs, and Google Earth, among other applications.

- These applications can support a large number of users simultaneously with HA.

- Notable technology achievements include the Google File System (GFS), MapReduce, BigTable, and Chubby.

- In 2008, Google announced the GAE web application platform which is becoming a common platform for many small cloud service providers.

- This platform specializes in supporting scalable (elastic) web applications.

- GAE enables users to run their applications on a large number of data centers associated with Google's search engine operations.

5.4.1 GAE Architecture

- Figure 5.4 shows the major building blocks of the Google cloud platform which has been used to deliver the cloud services highlighted earlier.

- GFS is used for storing large amounts of data.

- MapReduce is for use in application program development.

- Chubby is used for distributed application lock services.

- BigTable offers a storage service for accessing structured data.

- Users can interact with Google applications via the web interface provided by each application.

- Third-party application providers can use GAE to build cloud applications for providing services.

- The applications all run in data centers under tight management by Google engineers. Inside each data center, there are thousands of servers forming different clusters

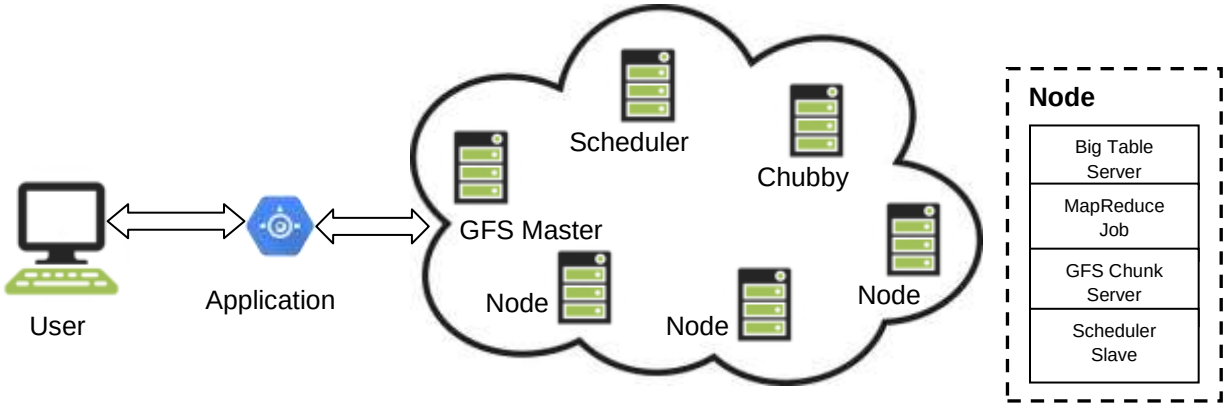


Figure 5.4 Google cloud platform

- Google is one of the larger cloud application providers, although it is fundamental service program is private and outside people cannot use the Google infrastructure to build their own service.

- The building blocks of Google's cloud computing application include the Google File System for storing large amounts of data, the MapReduce programming framework for application developers, Chubby for distributed application lock services, and BigTable as a storage service for accessing structural or semistructural data.

- With these building blocks, Google has built many cloud applications.

- Figure 5.4 shows the overall architecture of the Google cloud infrastructure.

- A typical cluster configuration can run the Google File System, MapReduce jobs and BigTable servers for structure data.

- Extra services such as Chubby for distributed locks can also run in the clusters.

- GAE runs the user program on Google's infrastructure. As it is a platform running third-party programs, application developers now do not need to worry about the maintenance of servers.

- GAE can be thought of as the combination of several software components.

- The frontend is an application framework which is similar to other web application frameworks such as ASP, J2EE and JSP.

- At the time of this writing, GAE supports Python and Java programming environments. The applications can run similar to web application containers.

- The frontend can be used as the dynamic web serving infrastructure which can provide the full support of common technologies.

5.4.2 Functional Modules of GAE

- The GAE platform comprises the following five major components.

- The GAE is not an infrastructure platform, but rather an application development platform for users.

    - The datastore offers object-oriented, distributed, structured data storage services based on BigTable techniques. The datastore secures data management operations.
    - The application runtime environment offers a platform for scalable web programming and execution. It supports two development languages: Python and Java.

- The software development kit (SDK) is used for local application development. The SDK allows users to execute test runs of local applications and upload application code.
- The administration console is used for easy management of user application development cycles, instead of for physical resource management.
- The GAE web service infrastructure provides special interfaces to guarantee flexible use and management of storage and network resources by GAE.

- Google offers essentially free GAE services to all Gmail account owners.

- The user can register for a GAE account or use your Gmail account name to sign up for the service.

- The service is free within a quota.

- If the user exceeds the quota, the page instructs how to pay for the service. Then the user can download the SDK and read the Python or Java guide to get started.

- Note that GAE only accepts Python, Ruby and Java programming languages.

- The platform does not provide any IaaS services, unlike Amazon, which offers IaaS and PaaS.

- This model allows the user to deploy user-built applications on top of the cloud infrastructure that are built using the programming languages and software tools supported by the provider (e.g., Java, Python).

- Azure does this similarly for .NET. The user does not manage the underlying cloud infrastructure.

- The cloud provider facilitates support of application development, testing, and operation support on a well-defined service platform.

5.4.3 GAE Applications

- Best-known GAE applications include the Google Search Engine, Google Docs, Google Earth and Gmail.
- These applications can support large numbers of users simultaneously.
- Users can interact with Google applications via the web interface provided by each application.
- Third party application providers can use GAE to build cloud applications for providing services.
- The applications are all run in the Google data centers.
- Inside each data center, there might be thousands of server nodes to form different clusters.
- Each cluster can run multipurpose servers.
- GAE supports many web applications.
- One is a storage service to store application specific data in the Google infrastructure.
- The data can be persistently stored in the backend storage server while still providing the facility for queries, sorting and even transactions similar to traditional database systems.

- GAE also provides Google specific services, such as the Gmail account service. This can eliminate the tedious work of building customized user management components in web applications.

5.5 Programming Environment for Google App Engine

- Several web resources (e.g., http://code.google.com/appengine/) and specific books and articles discuss how to program GAE.

- Figure 5.5 summarizes some key features of GAE programming model for two supported languages: Java and Python.

- A client environment that includes an Eclipse plug-in for Java allows you to debug your GAE on your local machine.

- Also, the GWT Google Web Toolkit is available for Java web application developers. Developers can use this, or any other language using a JVM based interpreter or compiler, such as JavaScript or Ruby.

- Python is often used with frameworks such as Django and CherryPy, but Google also supplies a built in webapp Python environment.

- There are several powerful constructs for storing and accessing data.

- The data store is a NOSQL data management system for entities that can be, at most, 1 MB in size and are labeled by a set of schema-less properties.

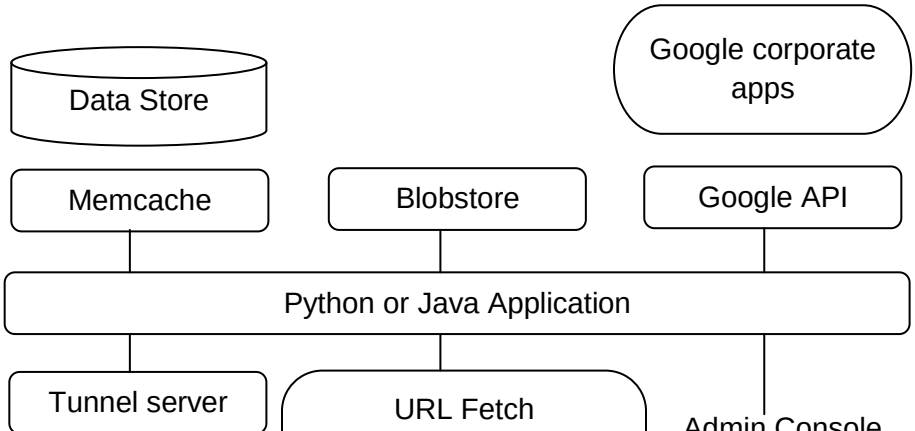


Figure 5.5 Programming Environment of Google AppEngine

- Queries can retrieve entities of a given kind filtered and sorted by the values of the properties.

- Java offers Java Data Object (JDO) and Java Persistence API (JPA) interfaces implemented by the open source Data Nucleus Access platform, while Python has a SQL-like query language called GQL.

- The data store is strongly consistent and uses optimistic concurrency control.

- An update of an entity occurs in a transaction that is retried a fixed number of times if other processes are trying to update the same entity simultaneously.

- The user application can execute multiple data store operations in a single transaction which either all succeed or all fail together.

- The data store implements transactions across its distributed network using entity groups.

- A transaction manipulates entities within a single group.
- Entities of the same group are stored together for efficient execution of transactions.
- The user GAE application can assign entities to groups when the entities are created.
- The performance of the data store can be enhanced by in-memory caching using the memcache, which can also be used independently of the data store.
- Recently, Google added the blobstore which is suitable for large files as its size limit is 2 GB.
- There are several mechanisms for incorporating external resources.
- The Google SDC Secure Data Connection can tunnel through the Internet and link your intranet to an external GAE application.
- The URL Fetch operation provides the ability for applications to fetch resources and communicate with other hosts over the Internet using HTTP and HTTPS requests.
- There is a specialized mail mechanism to send e-mail from your GAE application.
- Applications can access resources on the Internet, such as web services or other data, using GAE's URL fetch service.
- The URL fetch service retrieves web resources using the same high-speed Google infrastructure that retrieves web pages for many other Google products.

- There are dozens of Google “corporate” facilities including maps, sites, groups, calendar, docs, and YouTube, among others.

- These support the Google Data API which can be used inside GAE.

- An application can use Google Accounts for user authentication. Google Accounts handles user account creation and sign-in, and a user that already has a Google account (such as a Gmail account) can use that account with your app.

- GAE provides the ability to manipulate image data using a dedicated Images service which can resize, rotate, flip, crop and enhance images. An application can perform tasks outside of responding to web requests.

- A GAE application is configured to consume resources up to certain limits or quotas. With quotas, GAE ensures that your application would not exceed your budget and that other applications running on GAE would not impact the performance of your app.

- In particular, GAE use is free up to certain quotas.

- GFS was built primarily as the fundamental storage service for Google’s search engine.

- As the size of the web data that was crawled and saved was quite substantial, Google needed a distributed file system to redundantly store massive amounts of data on cheap and unreliable computers.

- In addition, GFS was designed for Google applications and Google applications were built for GFS.

- In traditional file system design, such a philosophy is not attractive, as there should be a clear interface between applications and the file system such as a POSIX interface.

- GFS typically will hold a large number of huge files, each 100 MB or larger, with files that are multiple GB in size quite common. Thus, Google has chosen its file data block size to be 64 MB instead of the 4 KB in typical traditional file systems.

- The I/O pattern in the Google application is also special.

- Files are typically written once, and the write operations are often the appending data blocks to the end of files.

- Multiple appending operations might be concurrent.

- BigTable was designed to provide a service for storing and retrieving structured and semi structured data.

- BigTable applications include storage of web pages, per-user data, and geographic locations.

- The scale of such data is incredibly large. There will be billions of URLs, and each URL can have many versions, with an average page size of about 20 KB per version.

- The user scale is also huge.

- There are hundreds of millions of users and there will be thousands of queries per second.

- The same scale occurs in the geographic data, which might consume more than 100 TB of disk space.

- It is not possible to solve such a large scale of structured or semi structured data using a commercial database system.

- This is one reason to rebuild the data management system and the resultant system can be applied across many projects for a low incremental cost.

- The other motivation for rebuilding the data management system is performance.

- Low level storage optimizations help increase performance significantly which is much harder to do when running on top of a traditional database layer.

- The design and implementation of the BigTable system has the following goals.

    - The applications want asynchronous processes to be continuously updating different pieces of data and want access to the most current data at all times.
    - The database needs to support very high read/write rates and the scale might be millions of operations per second.
    - The application may need to examine data changes over time.

- Thus, BigTable can be viewed as a distributed multilevel map. It provides a fault tolerant and persistent database as in a storage service.

- The BigTable system is scalable, which means the system has thousands of servers, terabytes of in-memory data, peta bytes of disk based data, millions of reads/writes per second and efficient scans.

- BigTable is a self managing system (i.e., servers can be added/removed dynamically and it features automatic load balancing).

- Chubby, Google's Distributed Lock Service Chubby is intended to provide a coarse-grained locking service.

- It can store small files inside Chubby storage which provides a simple namespace as a file system tree.

- The files stored in Chubby are quite small compared to the huge files in GFS.

5.6 OpenStack

- The OpenStack project is an open source cloud computing platform for all types of clouds, which aims to be simple to implement, massively scalable and feature rich.
- Developers and cloud computing technologists from around the world create the OpenStack project.
- OpenStack provides an Infrastructure as a Service (IaaS) solution through a set of interrelated services.
- Each service offers an application programming interface (API) that facilitates this integration.
- Depending on their needs, administrator can install some or all services.
- OpenStack began in 2010 as a joint project of Rackspace Hosting and NASA.
- As of 2012, it is managed by the OpenStack Foundation, a non-profit corporate entity established in September 2013 to promote OpenStack software and its community.
- Now, More than 500 companies have joined the project
- The OpenStack system consists of several key services that are separately installed.
- These services work together depending on your cloud needs and include the Compute, Identity, Networking, Image, Block Storage, Object Storage, Telemetry, Orchestration, and Database services.
- The administrator can install any of these projects separately and configure them standalone or as connected entities.

- Figure 5.6 shows the relationships among the OpenStack services:

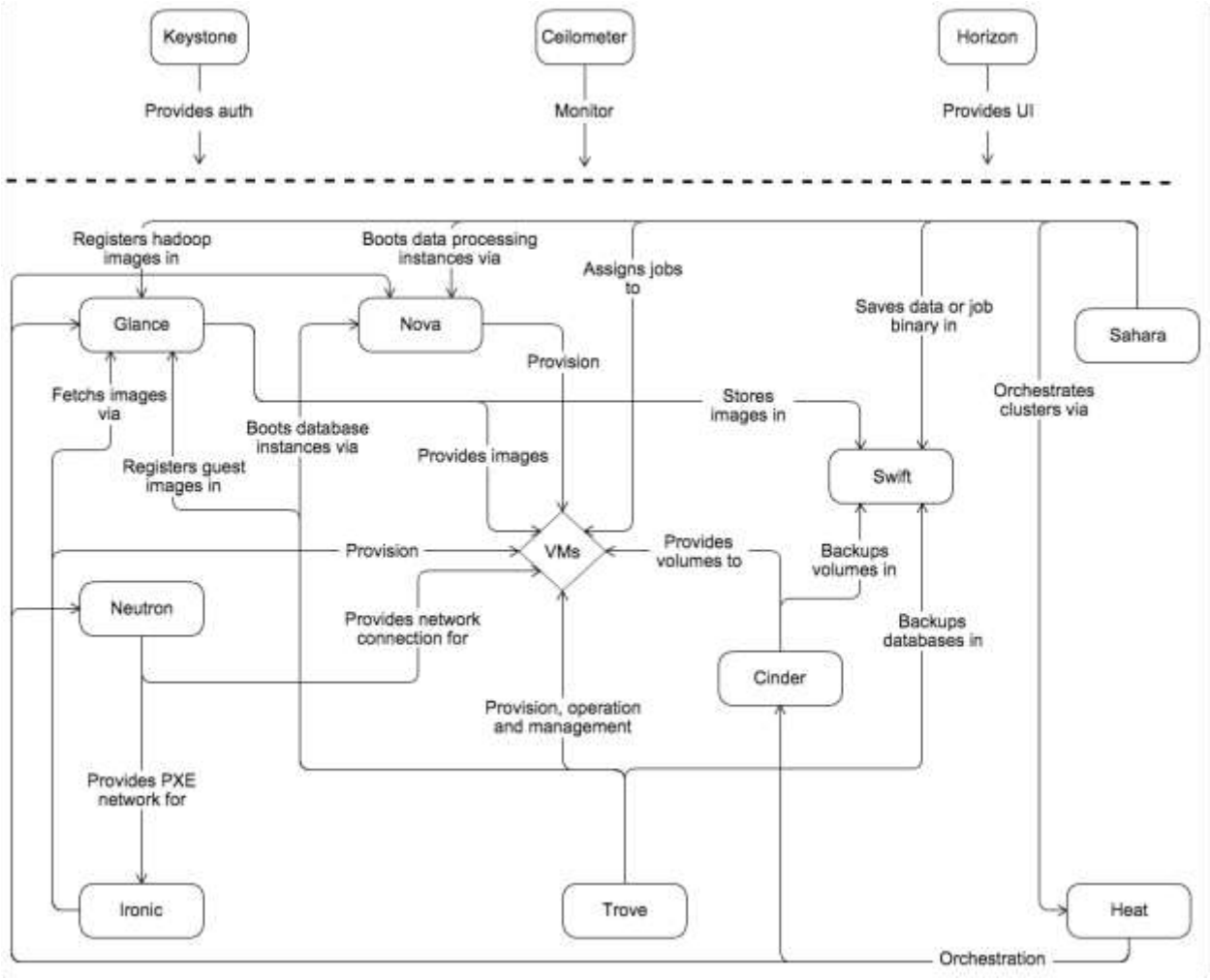


Figure 5.6 Relationship between OpenStack services

- To design, deploy, and configure OpenStack, administrators must understand the logical architecture.

- OpenStack consists of several independent parts, named the OpenStack services. All services authenticate through a common Identity service.

- Individual services interact with each other through public APIs, except where privileged administrator commands are necessary.

- Internally, OpenStack services are composed of several processes.

- All services have at least one API process, which listens for API requests, preprocesses them and passes them on to other parts of the service.

- With the exception of the Identity service, the actual work is done by distinct processes.

- For communication between the processes of one service, an AMQP message broker is used.

- The service's state is stored in a database.

- When deploying and configuring the OpenStack cloud, administrator can choose among several message broker and database solutions, such as RabbitMQ, MySQL, MariaDB, and SQLite.

- Users can access OpenStack via the web-based user interface implemented by the Horizon Dashboard, via command-line clients and by issuing API requests through tools like browser plug-ins or curl.

- For applications, several SDKs are available. Ultimately, all these access methods issue REST API calls to the various OpenStack services.

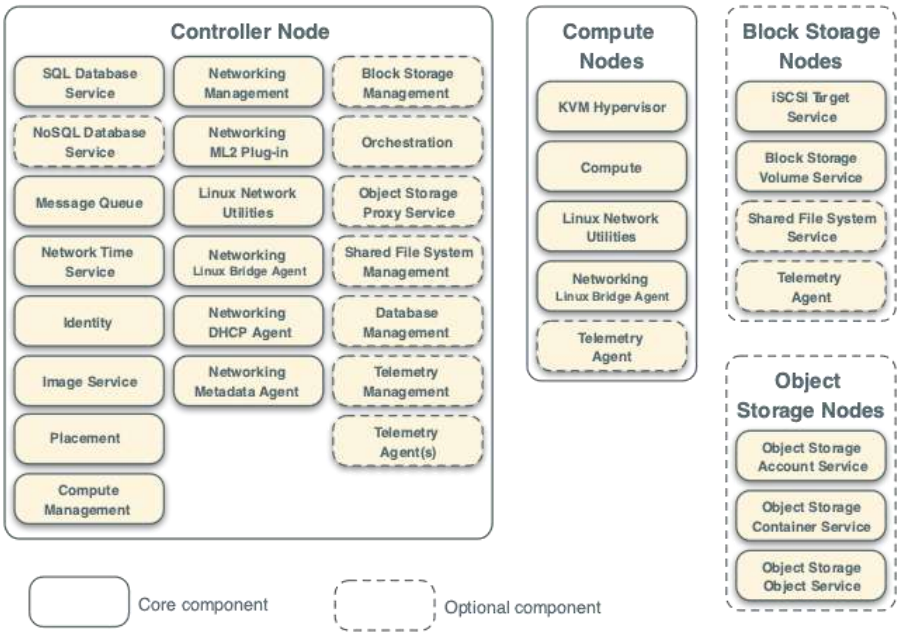


Figure 5.7 Example OpenStack architecture

- The controller node runs the Identity service, Image service, Placement service, management portions of Compute, management portion of Networking, various Networking agents, and the Dashboard.

- It also includes supporting services such as an SQL database, message queue, and NTP.

    - Optionally, the controller node runs portions of the Block Storage, Object Storage, Orchestration, and Telemetry services.

- The controller node requires a minimum of two network interfaces.

- The compute node runs the hypervisor portion of Compute that operates instances. By default, Compute uses the KVM hypervisor.

- The compute node also runs a Networking service agent that connects instances to virtual networks and provides firewalling services to instances via security groups.

- Administrator can deploy more than one compute node. Each node requires a minimum of two network interfaces.

- The optional Block Storage node contains the disks that the Block Storage and Shared File System services provision for instances.

- For simplicity, service traffic between compute nodes and this node uses the management network.

- Production environments should implement a separate storage network to increase performance and security.

- Administrator can deploy more than one block storage node. Each node requires a minimum of one network interface.

- The optional Object Storage node contains the disks that the Object Storage service uses for storing accounts, containers, and objects.

- For simplicity, service traffic between compute nodes and this node uses the management network.

- Production environments should implement a separate storage network to increase performance and security.

- This service requires two nodes. Each node requires a minimum of one network interface. Administrator can deploy more than two object storage nodes.

- The provider networks option deploys the OpenStack Networking service in the simplest way possible with primarily layer 2 (bridging/switching) services and VLAN segmentation of networks.

- Essentially, it bridges virtual networks to physical networks and relies on physical network infrastructure for layer-3 (routing) services.

- Additionally, a DHCP service provides IP address information to instances.

5.7 Federation in the Cloud

- One challenge in creating and managing a globally decentralized cloud computing environment is maintaining consistent connectivity between untrusted components while remaining fault tolerant.

- A key opportunity for the emerging cloud industry will be in defining a federated cloud ecosystem by connecting multiple cloud computing providers using a common standard.

- A notable research project being conducted by Microsoft called the Geneva Framework. This framework focuses on issues involved in cloud federation.

- Geneva has been described as claims based access platform and is said to help simplify access to applications and other systems.

- The concept allows for multiple providers to interact seamlessly with others and it enables developers to incorporate various authentication models that will work with any corporate identity system, including Active Directory,

- LDAPv3 based directories, application specific databases, and new user centric identity models such as LiveID, OpenID, and InfoCard systems.

- It also supports Microsoft's CardSpace and Novell's Digital Me.

- Federation in cloud is implemented by the use of Internet Engineering Task Force (IETF) standard Extensible Messaging and Presence Protocol (XMPP) and inter domain federation using the Jabber Extensible Communications Platform (Jabber XCP).

- Because this protocol is currently used by a wide range of existing services offered by providers as diverse as Google Talk, Live Journal, Earthlink, Facebook, ooVoo, Meebo, Twitter, the U.S. Marines Corps, the Defense Information Systems Agency (DISA), the U.S. Joint Forces Command (USJFCOM), and the National Weather Service.

- Session Initiation Protocol (SIP), which is the foundation of popular enterprise messaging systems such as IBM's Lotus Sametime and Microsoft's Live Communications Server (LCS) and Office Communications Server (OCS).

- Jabber XCP is a highly scalable, extensible, available, and device-agnostic presence solution built on XMPP and supports multiple protocols such as Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions (SIMPLE) and Instant Messaging and Presence Service (IMPS).

- Jabber XCP is a highly programmable platform, which makes it ideal for adding presence and messaging to existing applications or services and for building next-generation, presence based solutions.

- Over the last few years there has been a controversy brewing in web services architectures.

- Cloud services are being talked up as a fundamental shift in web architecture that promises to move us from interconnected silos to a collaborative network of services whose sum is greater than its parts.

- The problem is that the protocols powering current cloud services, SOAP (Simple Object Access Protocol) and a few other assorted HTTP based protocols, are all one-way information exchanges.

- Therefore cloud services are not real time, would not scale, and often cannot clear the firewall.

- Many believe that those barriers can be overcome by XMPP (also called Jabber) as the protocol that will fuel the Software as a Service (SaaS) models of tomorrow.

- Google, Apple, AOL, IBM, Live journal and Jive have all incorporated this protocol into their cloud based solutions in the last few years.

- Since the beginning of the Internet era, if the user wanted to synchronize services between two servers, the most common solution was to have the client “ping” the host at regular intervals, which is known as polling.

- Polling is how most of us check our email.

- XMPP’s profile has been steadily gaining since its inception as the protocol behind the open source instant messenger (IM) server jabberd in 1998.

- XMPP’s advantages include:

    - It is decentralized, meaning anyone may set up an XMPP server.
    - It is based on open standards.
    - It is mature multiple implementations of clients and servers exist.

- Robust security is supported via Simple Authentication and Security Layer (SASL) and Transport Layer Security (TLS).

- It is flexible and designed to be extended.

- XMPP is a good fit for cloud computing because it allows for easy two way communication

- XMPP eliminates the need for polling and focus on rich publish subscribe functionality

- It in it is XML-based and easily extensible, perfect for both new IM features and custom cloud services

- It is efficient and has been proven to scale to millions of concurrent users on a single service (such as Google's GTalk). And also it has a built-in worldwide federation model.

- Of course, XMPP is not the only pub-sub enabler getting a lot of interest from web application developers.

- An Amazon EC2-backed server can run Jetty and Cometd from Dojo.

- Unlike XMPP, Comet is based on HTTP and in conjunction with the Bayeux Protocol, uses JSON to exchange data.

- Given the current market penetration and extensive use of XMPP and XCP for federation in the cloud and that it is the dominant open protocol in that space.

- The ability to exchange data used for presence, messages, voice, video, files, notifications, etc., with people, devices and applications gain more power when they can be shared across organizations and with other service providers.

- Federation differs from peering, which requires a prior agreement between parties before a server-to-server (S2S) link can be established.

- In the past, peering was more common among traditional telecommunications providers (because of the high cost of transferring voice traffic).

- In the brave new Internet world, federation has become a de facto standard for most email systems because they are federated dynamically through Domain Name System (DNS) settings and server configurations.

5.8 Four Levels of Federation

- Federation is the ability for two XMPP servers in different domains to exchange XML stanzas.

- According to the XEP-0238: XMPP Protocol Flows for Inter-Domain Federation, there are at least four basic types of federation:

- Permissive federation

    - Permissive federation occurs when a server accepts a connection from a peer network server without verifying its identity using DNS lookups or certificate checking.
    - The lack of verification or authentication may lead to domain spoofing (the unauthorized use of a third-party domain name in an email message in order to pretend to be someone else), which opens the door to widespread spam and other abuses. With the release of the open source jabberd 1.2 server in October 2000, which included support for the Server Dialback protocol (fully supported in Jabber XCP), permissive federation met its demise on the XMPP network.

- Verified federation

    - This type of federation occurs when a server accepts a connection from a peer after the identity of the peer has been verified.
    - It uses information obtained via DNS and by means of domain-specific keys exchanged beforehand.
    - The connection is not encrypted, and the use of identity verification effectively prevents domain spoofing.
    - To make this work, federation requires proper DNS setup and that is still subject to DNS poisoning attacks.

- Verified federation has been the default service policy on the open XMPP since the release of the open-source jabberd 1.2 server.

- Encrypted federation

  - In this mode, a server accepts a connection from a peer if and only if the peer supports Transport Layer Security (TLS) as defined for XMPP in Request for Comments (RFC) 3920.
  - The peer must present a digital certificate.
  - The certificate may be self signed, but this prevents using mutual authentication.
  - If this is the case, both parties proceed to weakly verify identity using Server Dialback.
  - XEP-0220 defines the Server Dialback protocol, which is used between XMPP servers to provide identity verification.
  - Server Dialback uses the DNS as the basis for verifying identity
  - The basic approach is that when a receiving server receives a server-to-server connection request from an originating server, it does not accept the request until it has verified a key with an authoritative server for the domain asserted by the originating server.
  - Although Server Dialback does not provide strong authentication or trusted federation, and although it is subject to DNS poisoning attacks, it has effectively prevented most instances of address spoofing on the XMPP network since its release in 2000.
  - This results in an encrypted connection with weak identity verification.

- Trusted federation

  - In this federation, a server accepts a connection from a peer only under the stipulation that the peer supports TLS and the peer can present a digital certificate issued by a root certification authority (CA) that is trusted by the authenticating server.
  - The list of trusted root CAs may be determined by one or more factors, such as the operating system, XMPP server software or local service policy.

- In trusted federation, the use of digital certificates results not only in a channel encryption but also in strong authentication.
- The use of trusted domain certificates effectively prevents DNS poisoning attacks but makes federation more difficult, since such certificates have traditionally not been easy to obtain.

5.9 Federated Services and Applications

- S2S federation is a good start toward building a real-time communications cloud.
- Clouds typically consist of all the users, devices, services, and applications connected to the network.
- In order to fully leverage the capabilities of this cloud structure, a participant needs the ability to find other entities of interest.
- Such entities might be end users, multiuser chat rooms, real-time content feeds, user directories, data relays, messaging gateways, etc.
- Finding these entities is a process called discovery.
- XMPP uses service discovery (as defined in XEP-0030) to find the aforementioned entities.
- The discovery protocol enables any network participant to query another entity regarding its identity, capabilities and associated entities.
- When a participant connects to the network, it queries the authoritative server for its particular domain about the entities associated with that authoritative server.

- In response to a service discovery query, the authoritative server informs the inquirer about services hosted there and may also detail services that are available but hosted elsewhere.

- XMPP includes a method for maintaining personal lists of other entities, known as roster technology, which enables end users to keep track of various types of entities.

- Usually, these lists are comprised of other entities the users are interested in or interact with regularly.

- Most XMPP deployments include custom directories so that internal users of those services can easily find what they are looking for.

5.10 Future of Federation

- The implementation of federated communications is a precursor to building a seamless cloud that can interact with people, devices, information feeds, documents, application interfaces and other entities.

- The power of a federated, presence enabled communications infrastructure is that it enables software developers and service providers to build and deploy such applications without asking permission from a large, centralized communications operator.

- The process of server-to-server federation for the purpose of inter domain communication has played a large role in the success of XMPP, which relies on a small set of simple but powerful mechanisms for domain checking and security to generate verified, encrypted, and trusted connections between any two deployed servers.

- These mechanisms have provided a stable, secure foundation for growth of the XMPP network and similar real time technologies.

TWO MARK QUESTIONS

1. What is Hadoop?

- Hadoop is an open source implementation of MapReduce coded and released in Java (rather than C) by Apache.
- The Hadoop implementation of MapReduce uses the Hadoop Distributed File System (HDFS) as its underlying layer rather than GFS.

2. List the fundamental layers of Hadoop core.

- The Hadoop core is divided into two fundamental layers:
  - MapReduce engine
  - HDFS

3. Describe about HDFS.

- HDFS is a Hadoop distributed file system inspired by GFS that organizes files and stores their data on a distributed computing system.
- HDFS has a master/slave architecture containing a single NameNode as the master and a number of DataNodes as workers (slaves).
- To store a file in this architecture, HDFS splits the file into fixed-size blocks (e.g., 64 MB) and stores them on workers (DataNodes).
- The mapping of blocks to DataNodes is determined by the NameNode.

4. Is HDFS provides fault tolerant?

- One of the main aspects of HDFS is its fault tolerance characteristic. Since Hadoop is designed to be deployed on low-cost hardware by default, a hardware failure in this system is considered to be common rather than an exception.

5. List the issues to fulfill reliability requirements of the file system by hadoop.

- Block replication
- Replica placement
- Heartbeat and Block report messages

6. What is the purpose of heartbeat messages?

- Heartbeat is a periodic message sent to the NameNode by each DataNode in a cluster.

7. List the advantages of HDFS.

- The list of blocks per file will shrink as the size of individual blocks increases, and by keeping large amounts of data sequentially within a block, HDFS provides fast streaming reads of data.

8. Define MapReduce.

- The topmost layer of Hadoop is the MapReduce engine that manages the data flow and control flow of MapReduce jobs over distributed computing systems.
- Similar to HDFS, the MapReduce engine also has a master/slave architecture consisting of a single JobTracker as the master and a number of TaskTrackers as the slaves (workers).
- The JobTracker manages the MapReduce job over a cluster and is responsible for monitoring jobs and assigning tasks to TaskTrackers.
- The TaskTracker manages the execution of the map and/or reduce tasks on a single computation node in the cluster.

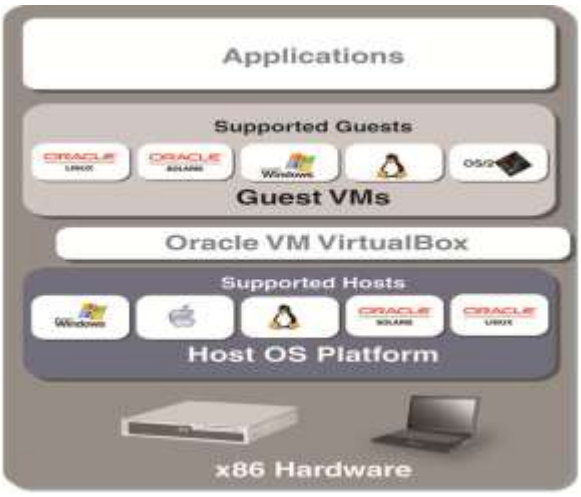
9. List the components contribute in running a job in Hadoop system.

- a user node
- a JobTracker
- TaskTrackers

10. What is the use of VirtualBox?

- Oracle VM VirtualBox is a cross-platform virtualization application.
- For one thing, it installs on the existing Intel or AMD-based computers, whether they are running Windows, Mac OS X, Linux, or Oracle Solaris operating systems (OSes).
- Secondly, it extends the capabilities of existing computer so that it can run multiple OSes, inside multiple virtual machines, at the same time.

11. Illustrate the architecture of VirtualBox.



12. List the three disk image formats used in VirtualBox:

- VDI: This format is the VirtualBox-specific VirtualBox Disk Image and stores data in files bearing a ".vdi".
- VMDK: This open format is used by VMware products and stores data in one or more files bearing ".vmdk" filename extensions.
- VHD: This format is used by Windows Virtual PC and Hyper-V, and is the native virtual disk format of the Microsoft Windows operating system.

13. Describe about GAE.

- Google's App Engine (GAE) which offers a PaaS platform supporting various cloud and web applications.

- This platform specializes in supporting scalable (elastic) web applications.
- GAE enables users to run their applications on a large number of data centers associated with Google's search engine operations.

14. Mention the components maintained in a node of Google cloud platform.

- GFS is used for storing large amounts of data.
- MapReduce is for use in application program development.
- Chubby is used for distributed application lock services.
- BigTable offers a storage service for accessing structured data.

15. List the functional modules of GAE.

- Datastore
- Application runtime environment
- Software development kit (SDK)
- Administration console
- GAE web service infrastructure

16. List the applications of GAE.

- Well-known GAE applications include the Google Search Engine, Google Docs, Google Earth, and Gmail.
- These applications can support large numbers of users simultaneously.
- Users can interact with Google applications via the web interface provided by each application.
- Third-party application providers can use GAE to build cloud applications for providing services.

17. Mention the goals for design and implementation of the BigTable system.

- The applications want asynchronous processes to be continuously updating different pieces of data and want access to the most current data at all times.

- The database needs to support very high read/write rates and the scale might be millions of operations per second.
- The application may need to examine data changes over time.

18. Describe about Openstack.

- The OpenStack project is an open source cloud computing platform for all types of clouds, which aims to be simple to implement, massively scalable, and feature rich.
- Developers and cloud computing technologists from around the world create the OpenStack project.
- OpenStack provides an Infrastructure-as-a-Service (IaaS) solution through a set of interrelated services.

19. List the key services of OpenStack.

- The OpenStack system consists of several key services that are separately installed.
- Compute, Identity, Networking, Image, Block Storage, Object Storage, Telemetry, Orchestration and Database services.

20. What is the need of federated cloud ecosystem?

- One challenge in creating and managing a globally decentralized cloud computing environment is maintaining consistent connectivity between untrusted components while remaining fault-tolerant.
- A key opportunity for the emerging cloud industry will be in defining a federated cloud ecosystem by connecting multiple cloud computing providers using a common standard.
- A notable research project being conducted by Microsoft, called the Geneva Framework, focuses on issues involved in cloud federation.

21. List the advantages of Extensible Messaging and Presence Protocol.

- XMPP's is decentralized, meaning anyone may set up an XMPP server. It is based on open standards. It is mature multiple implementations of clients and servers exist.

22. List the levels of Federation.

- Permissive federation
- Verified federation
- Encrypted federation
- Trusted federation

23. What is S2S federation?

- S2S federation is a good start toward building a real-time communications cloud. Clouds typically consist of all the users, devices, services, and applications connected to the network.

24. What is the future of federation?

- The power of a federated, presence enabled communications infrastructure is that it enables software developers and service providers to build and deploy such applications without asking permission from a large, centralized communications operator.
- These mechanisms have provided a stable, secure foundation for growth of the XMPP network and similar real time technologies.