Software Defined Networking

Software-Defined Networking (SDN) is a networking architecture that separates the control plane from the data plane and centralizes the network controller. In the conventional network architecture the control plane and data plane are coupled.

The limitations of the conventional network architectures are as follows:

Complex Network Devices: Conventional networks are getting increasingly complex with more and more protocols being implemented to improve link speeds and reliability.

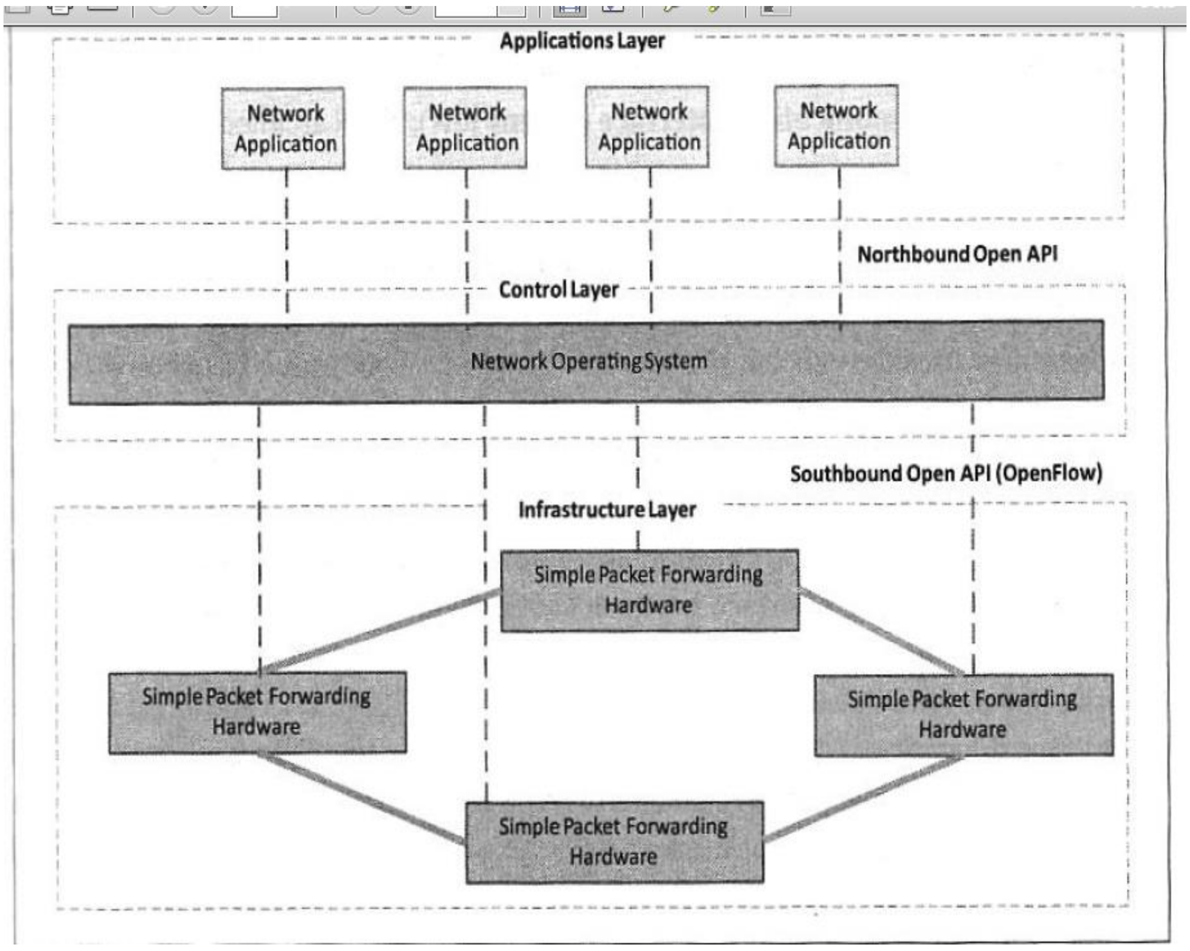
Management Overhead: Conventional networks involve significant management overhead. Network managers find it increasingly difficult to manage multiple network devices and interfaces from multiple vendors.

Limited Scalability: The virtualization technologies used in cloud computing environments has increased the number of virtual hosts requiring network access.

SDN attempts to create network architectures that are simpler, inexpensive, scalable, agile and easy to manage.

Components of SDN:
SDN is comprised of three key components: the data plane, the control plane, and the application layer. The data plane is responsible for forwarding network traffic, while the control plane manages network infrastructure and makes decisions about how network traffic should be handled. The application layer consists of software applications that run on top of the SDN infrastructure.

This figures shows the SDN architecture and the SDN layers in which the control and data planes are decoupled and the network controller is centralized.



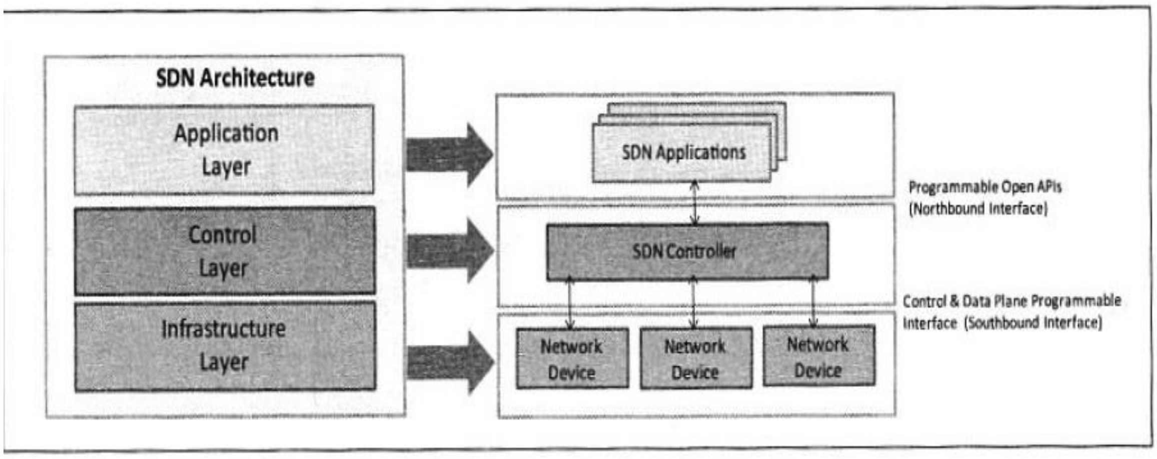


Figure 2.13: SDN layers

The key elements of SDN architecture as follows

Centralized Network Controller: With decoupled the control and data planes and centralized network controller, the network administrators can rapidly configure the network.

Programmable Open APIs: SDN architecture supports programmable open APIs for interface between the SDN application and control layers (Northbound interface).

Standard Communication Interface (OpenFlow): SDN architecture uses a standard communication interface between the control and infrastructure layers (Southbound interface).

UNIT-2

Compute Services

Compute services provide dynamically scalable compute capacity in the cloud. Compute resources can be provisioned on-demand in the form of virtual machines. Compute services can be accessed from the web consoles of these services that provide graphical user interfaces for provisioning, managing and monitoring these services.

Features

Scalable: Compute services allow rapidly provisioning as many virtual machine instances as required. The provisioned capacity can be scaled-up or down based on the workload levels.

Flexible: Compute services give a wide range of options for virtual machines with multiple instance types, operating systems, zones/regions, etc.

Secure: Compute services provide various security features that control the access to the virtual machine instances such as security groups, access control lists, network fire- walls, etc. Users can securely connect to the instances by using authentication mechanisms.

Cost effective: Cloud service providers offer various billing options such as on- demand instances which are billed per-hour, reserved instances which are reserved after one-time initial payment, spot instances for which users can place bids, etc. The examples are

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud (EC2) is a compute service provided by Amazon. To launch a new instance click on the launch instance button. This will open a wizard where you can select the Amazon machine image (AMI) with which you want to launch the instance. You can also create their own AMIS with custom applications, libraries and data. Instances can be launched with a variety of operating systems. When you launch an instance you specify the instance type (micro, small, medium, large, extra-large, etc.) the number of instances to launch based on the selected AMI and availability zones for the instances.

Google Compute Engine

Google Compute Engine is a compute service provided by Google. GCE console allows users to create and manage compute instances. To create a new instance, the user selects an instance machine type, a zone in which the instance will be launched, a machine image for the instance and provides an instance name, instance tags and meta-data. Every instance is launched with a disk resource.

Windows Azure Virtual Machines

Windows Azure Virtual Machines is the compute service from Microsoft. To create a new instance, you select the instance type and the machine image.

Storage Services

Cloud storage services allow storage and retrieval of any amount of data, at anytime from anywhere on the web. Most cloud storage services organize data into buckets or containers. Buckets or containers store objects which are individual pieces of data.

Features

Scalability: Cloud storage services provide high capacity and scalability. Objects up to several tera-bytes in size can be uploaded and multiple buckets/containers can be created on cloud storages.

Replication: When an object is uploaded it is replicated at multiple facilities and/or on multiple devices within each facility.

Access Policies: Cloud storage services provide several security features such as Access Control Lists (ACLs), bucket/container level policies, etc. ACLs can be used to selectively grant access permissions on individual objects

Encryption: Cloud storage services provide Server Side Encryption (SSE) options to encrypt all data stored in the cloud storage.

Consistency: Strong data consistency is provided for all upload and delete operations. Therefore, any object that is uploaded can be immediately downloaded after the upload is complete. The examples are

Amazon Simple Storage Service

Amazon Simple Storage Service(S3) is an online cloud-based data storage infrastructure for storing and retrieving any amount of data. S3 provides highly reliable, scalable, fast, fully redundant and affordable storage infrastructure. You must create a bucket before you can store data on S3..

Google Cloud Storage

Objects in GCS are organized into buckets. ACLs are used to control access to objects and buckets. ACLS can be configured to share objects and buckets with the entire world, a Google group, a Google-hosted domain, or specific Google account holders.

Windows Azure Storage

Windows Azure Storage is the cloud storage service from Microsoft. Windows Azure Storage provides various storage services such as blob storage service, table service and queue service. Blobs are organized into containers. Two kinds of blobs can be stored block blobs and page blobs

Database Services

Cloud database services allow you to set-up and operate relational or non-relational databases in the cloud. The benefit of using cloud database services is that it relieves the application developers from the time-consuming database administration tasks. Popular relational databases provided by various cloud service providers include MySQL, Oracle, SQL Server, etc. The non-relational (No-SQL) databases provided by cloud service providers are mostly proprietary solutions.

Features

Scalability: Cloud database services allow provisioning as much compute and storage resources as required to meet the application workload levels. Provisioned capacity can be scaled-up or down.

Reliability: Cloud database services are reliable and provide automated backup and snapshot options.

Performance: Cloud database services provide guaranteed performance with options such as guaranteed input/output operations per second (IOPS) which can be provisioned upfront.

Security: Cloud database services provide several security features to restrict the access to the database instances and stored data, such as network firewalls and authentication mechanisms.

The examples are

1. Amazon Relational Data Store

Amazon Relational Database Service (RDS) is a web service that makes it easy to setup, operate and scale a relational database in the cloud. The Amazon console provides an instance launch wizard that allows you to select the type of database to create (MySQL, Oracle or SQL Server) database instance size, allocated storage, DB instance identifier, DB username and password.

Amazon DynamoDB

Amazon DynamoDB is the non-relational (No-SQL) database service from Amazon. The DynamoDB data model includes include tables, items and attributes.

2. Google Cloud SQL

Google SQL is the relational database service from Google. Google Cloud SQL service allows you to host MySQL databases in the Google's cloud. You can create new database instances from the console and manage existing instances.

Google Cloud Datastore

Google Cloud Datastore is a fully managed non-relational database from Google. Cloud Datastore offers ACID transactions and high availability of reads and writes.

3. Windows Azure SQL Database

Windows Azure SQL Database is the relational database service from Microsoft. Azure SQL Database is based on the SQL server, but it does not give each customer a separate instance of SQL server

Windows Azure Table Service

Windows Azure Table Service is a non-relational (No-SQL) database service from Microsoft. The Azure Table Service data model consists of tables having multiple entities.

Application Services

The cloud application services such as application runtimes and frameworks, queuing services, email services, notification services and media services. The features of Application services are

Application Runtimes & Frameworks

Cloud-based application runtimes and frameworks allow developers to develop and host applications in the cloud. Application runtimes provide support for programming languages e.g. Java, Python, or Ruby.

Runtimes: App Engine provides runtime environments for Java, Python, PHP and Go programming language.

Sandbox: The sandbox environment provides a limited access to the underlying operating system. The sandbox environment allows App Engine to distribute web requests for the application across multiple servers.

Web Frameworks: App Engine provides a simple Python web application framework called webapp2.

Datastore: App Engine provides a no-SQL data storage service.

Authentication: App Engine applications can be integrated with Google Accounts for user authentication.

URL Fetch service: URL Fetch service allows applications to access resources on the Internet, such as web services or other data.

Email service: Email service allows applications to send email messages. The Amazon Email and Google mail services are different types of services.

Image Manipulation service: Image Manipulation service allows applications to resize, crop, rotate, flip and enhance images.

Memcache: Memcache service is a high-performance in-memory key-value cache service that applications can use for caching data items that do not need a persistent storage.

Task Queues: Task queues allow applications to do work in the background by breaking up work into small, discrete units, called tasks which are enqueued in task queues.

Scheduled Tasks service: App Engine provides a Cron service for scheduled tasks that trigger events at specified times and regular intervals. This service allows applica- tions to perform tasks at defined times or regular intervals.

Queuing Services

Cloud-based queuing services allow de-coupling application components. The de-coupled components communicate via messaging queues. Queues are useful for asynchronous pro- cessing

Notification Services

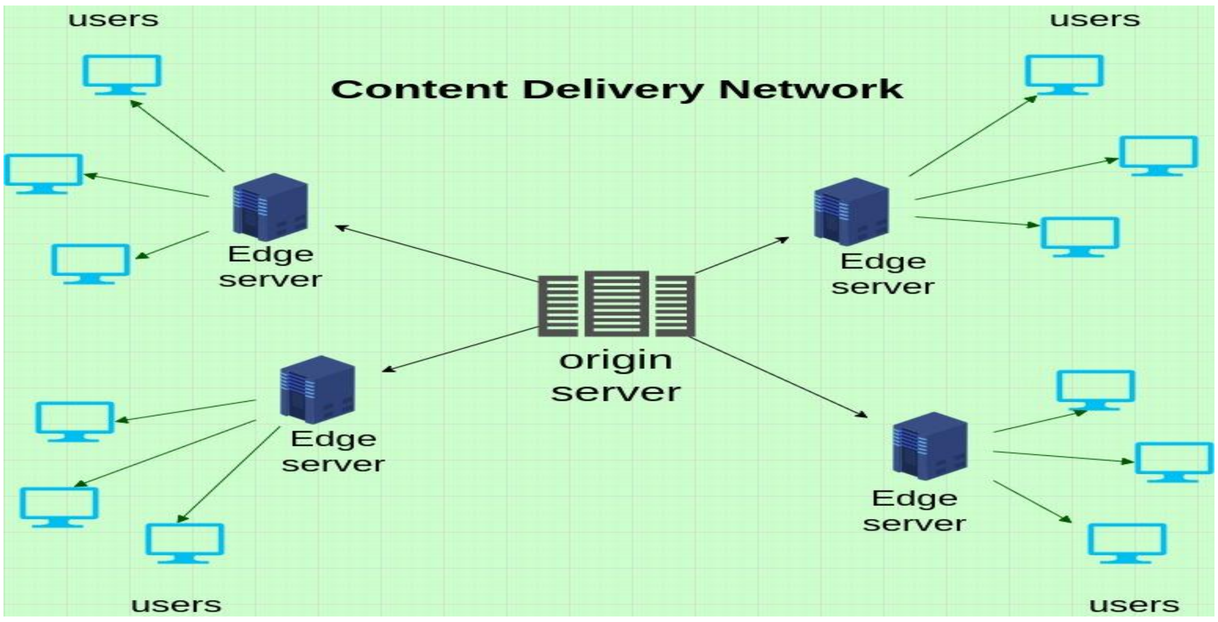
Cloud-based notification services or push messaging services allow applications to push messages to internet connected smart devices such as smart phones, tablets, etc. Push messaging services are based on publish-subscribe model. The Amazon Simple Notification Service is provided by Amazon. Windows Azure Notification Hubs provided by Azure.

Media Services

Cloud service providers provide various types of media services that can be used by applications for manipulating, transforming or transcoding media such as images, videos, etc.

Content Delivery Services

Cloud-based content delivery service include Content Delivery Networks (CDNs). A CDN is a distributed system of servers located across multiple geographic locations to serve content to end-users with high availability and high performance. CDNs are useful for serving static content such as text, images, scripts, etc., and streaming media. CDNs have a number of edge locations deployed in multiple locations, often over multiple backbones.



The examples are

Amazon CloudFront

Amazon CloudFront is a content delivery service from Amazon. CloudFront can be used to deliver dynamic, static and streaming content using a global network of edge locations. The content in CloudFront is organized into distributions. Each distribution specifies the original location of the content to be delivered which can be an Amazon S3 bucket, an Amazon EC2 instance, or an Elastic Load Balancer, or your own origin server. Distributions can be accessed by their domain names.

Windows Azure Content Delivery Network

Windows Azure Content Delivery Network (CDN) is the content delivery service from Microsoft. Azure CDN caches Windows Azure blobs and static content at the edge locations to improve the performance of web sites.

Analytics Services

Cloud-based analytics services allow analyzing massive data sets stored in the cloud either in cloud storages or in cloud databases using programming models such as MapReduce. Using cloud analytics services applications can perform data-intensive tasks such as such as data mining, log file analysis, machine learning, web indexing, etc.

Amazon Elastic MapReduce

Amazon Elastic MapReduce is the MapReduce service from Amazon based the Hadoop framework. It Supports various job types:

- Custom JAR: Custom IAR job flow runs a Java program that you have uploaded to Amazon S3.
- Hive program: Hive is a data warehouse system for Hadoop. You can use Hive to process data using the SQL-like language, called Hive-QL.
- Streaming job: Streaming job flow runs a single Hadoop job consisting of map and reduce functions implemented in a script or binary that you have uploaded to Amazon.

some of the examples are

Google MapReduce Service

Google MapReduce Service is a part of the App Engine platform. App Engine MapReduce is optimized for App Engine environment and provides capabilities such as automatic sharding for faster execution, standard data input readers for iterating over blob and datastore data, standard output writers, etc.

Google BigQuery

Google BigQuery is a service for querying massive datasets. BigQuery allows querying datasets using SQL-like queries. To query data, it is first loaded into BigQuery using the BigQuery console or BigQuery command line tool or BigQuery API.

Windows Azure HDInsight

Windows Azure HDInsight is an analytics service from Microsoft. HDInsight deploys and provisions Hadoop clusters in the Azure cloud and makes Hadoop available as a service.

Deployment & Management Services

Cloud-based deployment & management services allow you to easily deploy and manage applications in the cloud. These services automatically handle deployment tasks such as capacity provisioning, load balancing, auto-scaling, and application health monitoring. Some of the examples are

Amazon Elastic Beanstalk

Amazon provides a deployment service called Elastic Beanstalk that allows you to quickly deploy and manage applications in the AWS cloud. Elastic Beanstalk supports Java, PHP, .NET, Node.js, Python, and Ruby applications. With Elastic Beanstalk you just need to upload the application and specify configuration settings in a simple wizard and the service automatically handles instance provisioning, server configuration, load balancing and monitoring.

Amazon CloudFormation

Amazon CloudFormation is a deployment management service from Amazon. With Cloud- Front you can create deployments from a collection of AWS resources such as Amazon Elastic Compute Cloud, Amazon Elastic Block Store, Amazon Simple Notification Service, Elastic Load Balancing and Auto Scaling. CloudFormation stacks are created from Cloud- Formation templates. You can create your own templates or use the predefined templates.

Identity & Access Management Services

Identity & Access Management (IDAM) services allow managing the authentication and authorization of users to provide secure access to cloud resources. IDAM services are useful for organizations which have multiple users who access the cloud resources. Using IDAM services you can manage user identifiers, user permissions, security credentials and access keys.

Amazon Identity & Access Management

AWS Identity and Access Management (IAM) allows you to manage users and user permissions for an AWS account. With IAM you can manage users, security credentials such as access keys, and permissions that control which AWS resources users can access.

Windows Azure Active Directory

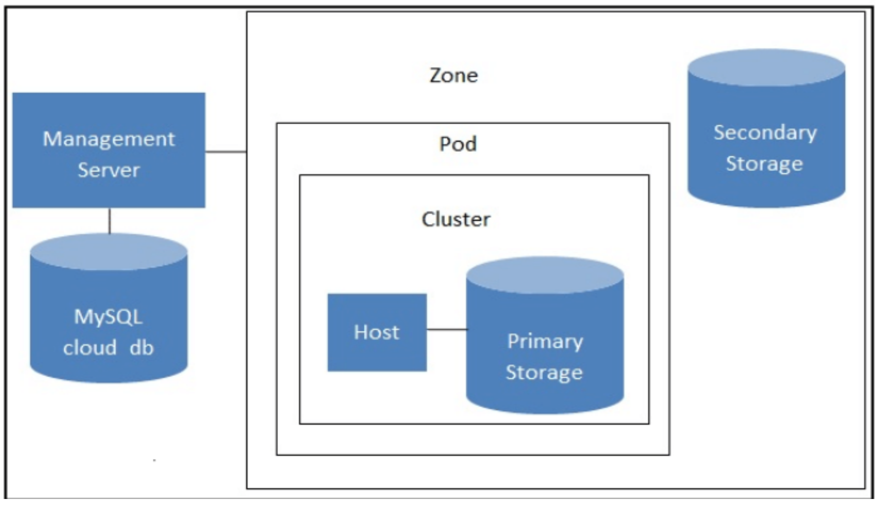
Windows Azure Active Directory is an Identity & Access Management Service from Microsoft. Azure Active Directory provides a cloud-based identity provider that easily integrates with your on-premises active directory deployments and also provides support for third party identity providers.

Open Source Private Cloud Software

In this we covers open source cloud software that can be used to build private clouds.

1. **CloudStack**

Apache CloudStack is an open source cloud software that can be used for creating private cloud. CloudStack manages the network, storage, and compute nodes that make up a cloud infrastructure. This figure shows the architecture of CloudStack which is basically the Management Server.



The Management Server: manages one or more zones where each zone is typically a single datacenter. Each zone has one or more pods.

pod: pod is a rack of hardware comprising of a switch and one or more clusters.

cluster:A cluster consists of one or more hosts and a primary storage.

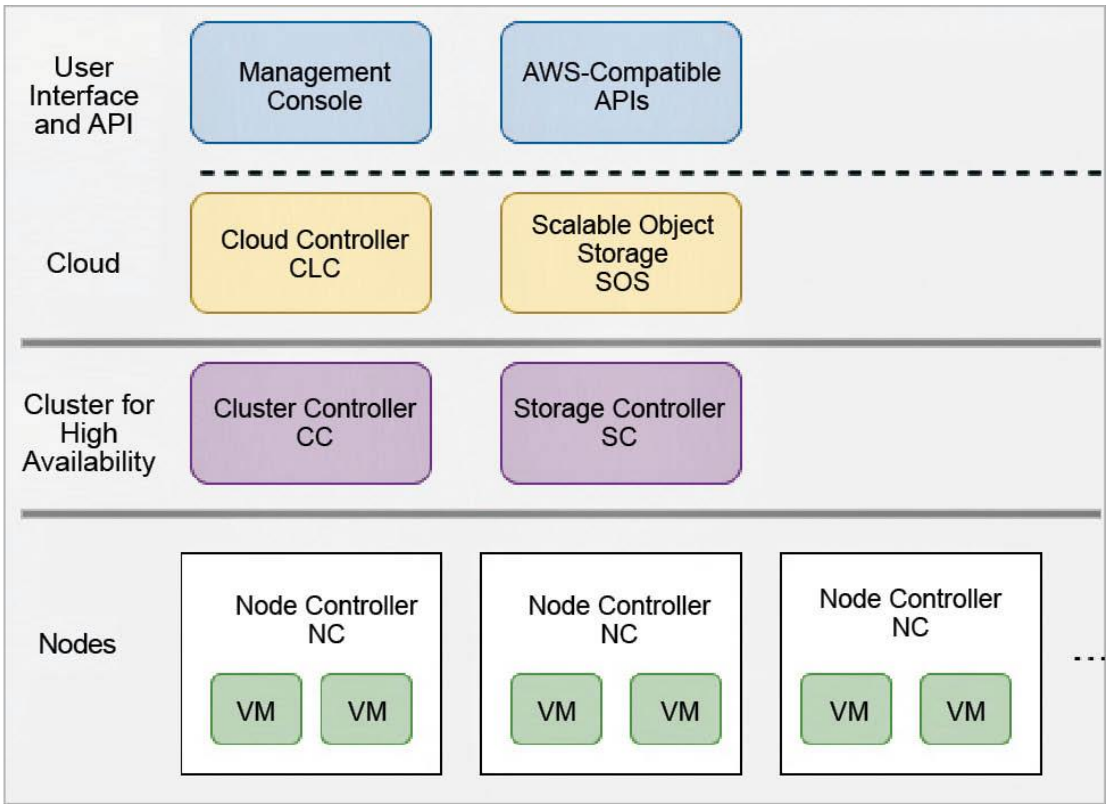
host: A host is a compute node that runs guest virtual machines.

primary storage :The primary storage of a cluster stores the disk volumes for all the virtual machines running on the hosts in that cluster.

secondary storage: It stores templates, ISO images, and disk volume snapshots.

2. Eucalyptus

Eucalyptus is an open source private cloud software for building private and hybrid clouds that are compatible with Amazon Web Services (AWS). This figure shows the architecture of Eucalyptus.



Node Controller (NC): It hosts the virtual machine instances and manages the virtual network endpoints.

Cluster controller(CC): The CC manages the virtual machines and is the front-end for a cluster.

Storage controller(SC): The SC manages the Eucalyptus block volumes and snapshots to the instances within its specific cluster.

Cloud Controller (CLC):CLC provides an administrative interface for cloud management and performs high-level resource scheduling.

Walrus:It serves as a persistent storage to all of the virtual machines in the Eucalyptus cloud.

3. OpenStack

OpenStack is a cloud operating system comprising of a collection of interacting services that control computing, storage, and networking resources. This figure shows the architecture of OpenStack.

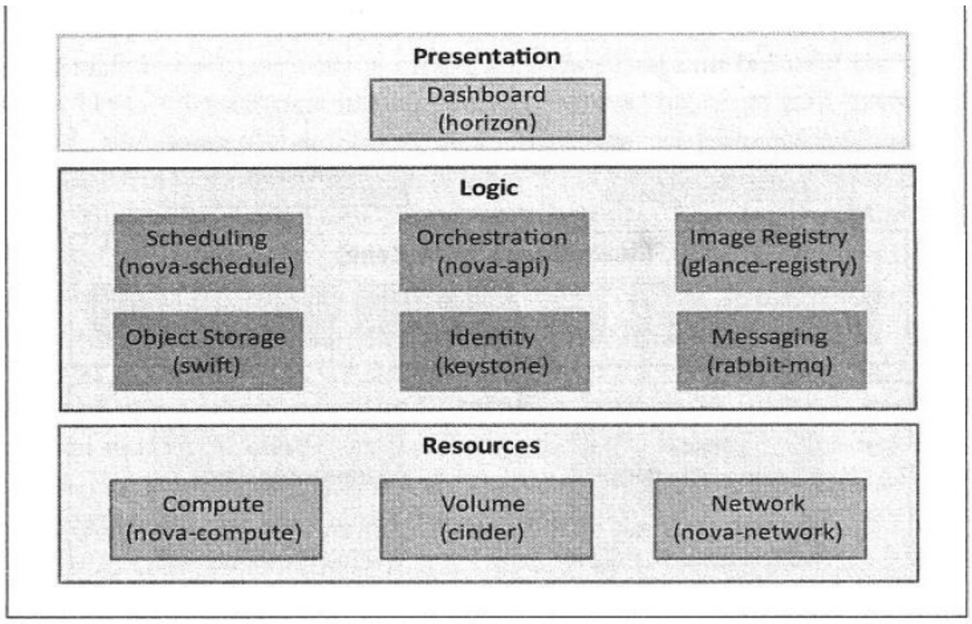


Figure 3.23: OpenStack architecture

Compute (Nova): Compute is a controller that is used to manage resources in virtualized environments.

Object Storage (Swift): To store and retrieve arbitrary data in the cloud, object storage is used. In Swift, it is possible to store the files, objects, backups, images, videos, virtual machines, and other unstructured data.

Block Storage (Cinder): Cinder manages to add, remove, create new disk space in the server. This component provides the virtual storage for the virtual machines in the system.

Dashboard (Horizon): This is the first component that the user sees in the OpenStack. Through individual API (Application programming interface), developers can access the OpenStack's components, but through the dashboard, system administrators can look at what is going on in the cloud and manage it as per their need.

Identity Service (Keystone): It is the central repository of all the users and their permissions for the OpenStack services they use.

Orchestration (Heat): It allows the developers to store the cloud application's necessities as a file so that all-important resources are available in handy.

Introduction

Web applications have evolved significantly in the past decade and are now quite dynamic in nature allowing users to interact and collaborate, include user generated content such as comments and discussions, integrate social networks and multimodal content such as text, images, audio, video, presentations, etc., in various formats. Due to this dynamic nature of modern web applications, the traffic patterns for such applications are becoming more and more unpredictable.

Design Considerations for Cloud Applications

In this we discuss about the important design considerations for developing applications that can leverage the benefits of cloud computing.

1. Scalability

Scalability is an important factor that drives the application designers to move to cloud computing environments. With the growth of cloud computing application designers can provision adequate resources to meet their workload levels

Loose coupling of components: Traditional applications used Tightly coupled components use procedure based tight coupling and hard-wired links which make it difficult to scale application components independently. By designing loosely coupled components, it is possible to scale each component independently.

Asynchronous communication: By allowing asynchronous communication between components, it is possible to add capacity by adding additional servers when the application load increases.

Stateless design: Stateless designs that store state outside of the components in a separate database allow scaling the application components independently.

Database choice and design: Choice of the database and the design of data storage schemes affect the application scalability, Decisions such as whether to choose a traditional relational database (SQL approach) with strict schemas or a schema-less database (No-SQL approach) should be made after careful analysis of the application's data storage and analysis requirements.

2. Reliability & Availability

Reliability of a system is defined as the probability that a system will perform the intended functions under stated conditions for a specified amount of time. Availability is the probability that a system will perform a specified function under given conditions at a prescribed time.

The important considerations to be kept in mind are

No single point of failure: Traditional application design approaches which have single points of failure such as a single database server or a single application server have the risk of complete breakdowns in case the of failure of the critical resource.

Trigger automated actions on failures: By using failures and triggers for automated actions it is possible to improve the application reliability and availability. For example, if an application server experiences high CPU usage and is a unable to server new requests, a new application server is automatically launched.

Logging: Logging all events in all the application components can help in detecting bottlenecks and failures so that necessary design/deployment changes can be made to improve application reliability and availability.

Replication: All application data should be replicated. Replication is used to create and maintain multiple copies of the data in the cloud. In the event of data loss at the primary location, organizations can continue to operate their applications from secondary data sources.

3. Security

Security is an important design consideration for cloud applications given the out- sourced nature of cloud computing environments. Key security considerations for cloud computing environments include:

- Securing data at rest
- Securing data in motion
- Authentication
- Authorization
- Identity and access management
- Key management
- Data integrity
- Auditing

4. Maintenance & Upgradation

To achieve a rapid time-to-market, businesses typically launch their applications with a core set of features ready and then incrementally add new features as and when they are complete. Businesses may need to adapt their applications based on the feedback from the users. In such scenarios, it is important to design applications with low maintenance and upgradation costs.

5. Performance

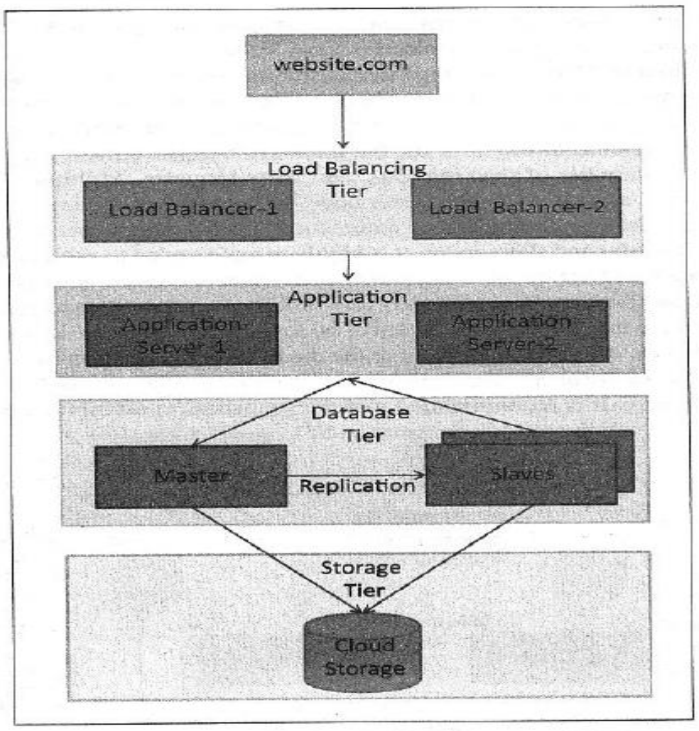
Applications should be designed while keeping the performance requirements in mind. Performance requirements depend on the type of the application. For example applications which experience high database read-intensive workloads, can benefit from read-replication or caching approaches.

Reference Architectures for Cloud Applications

Multi-tier cloud applications can have various deployment architecture alternatives. Choosing the right deployment architecture is important to ensure that the application meets the specified performance requirements.

Deployment architecture for e-Commerce

This figure shows a typical deployment architecture for e-Commerce, Business-to-Business, Banking and Financial applications. The various tiers in this deployment include:



'ypical deployment architecture for e-Commerce, Business-to-Busines

- **Load Balancing Tier:** The first tier is the load balancing tier. Load balancing tier consists of one or more load balancers. It is recommended to have at least two load balancer instances to avoid the single point of failure.
- **Application Tier:** The second tier is the application tier that consists of one or more application servers. For this tier, it is recommended to configure auto scaling. Auto scaling can be triggered when the recorded values for any of the specified metrics such as CPU usage, memory usage, etc. goes above defined thresholds.

- **Database Tier:** The third tier is the database tier which includes a master database instance and multiple slave instances. The master node serves all the write requests and the read requests are served from the slave nodes. This improves the throughput for the database tier since most applications have a higher number of read requests than write requests. Multiple slave nodes also serve as a backup for the master node. In the event of failure of the master node, one of the slave nodes can be automatically configured to become the master.

<u>Deployment architecture for content delivery applications</u>

This figure shows the typical deployment architecture for content delivery applications such as online photo albums, video webcasting, etc.

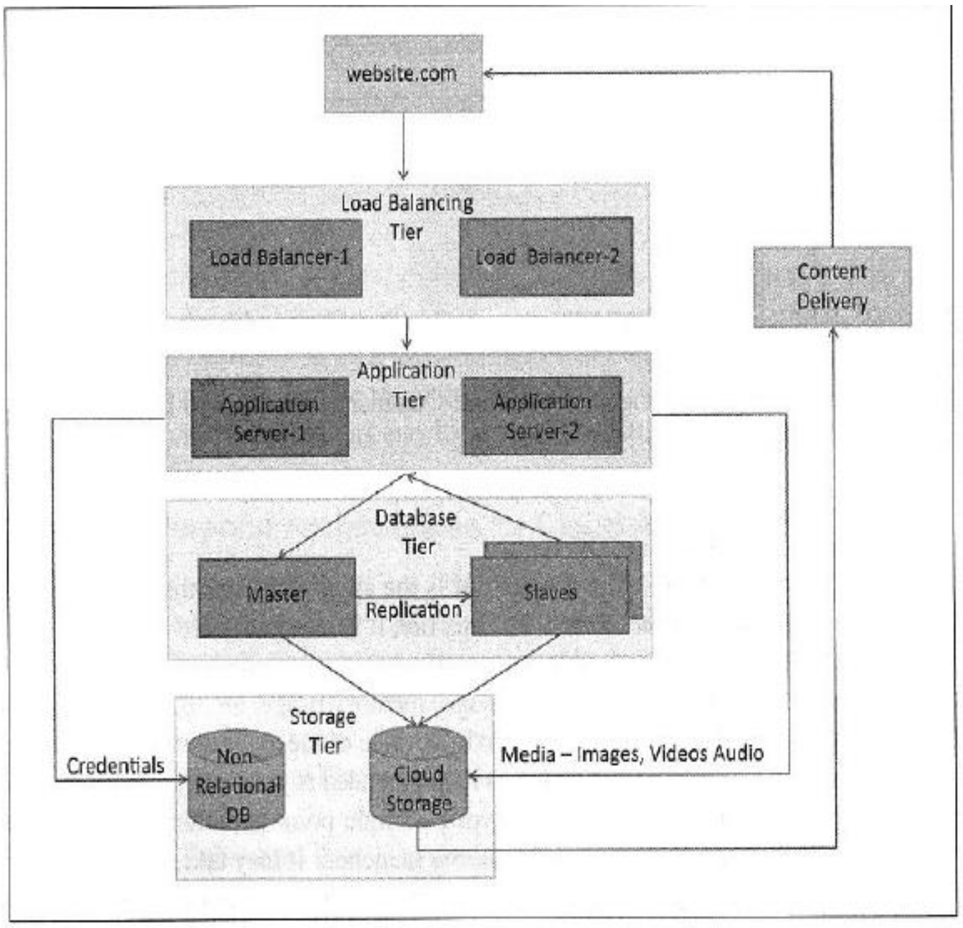


Figure 5.2: Typical deployment architecture for content delivery applications such as online photo albums, video webcasting, etc.

It shows a typical deployment architecture for content delivery applications such as online photo albums, video webcasting, etc. Both relational and non-relational data stores are shown in this deployment. A content delivery network (CDN) which consists of a global network of edge locations is used for media delivery. CDN is used to speed up the delivery of static content such as images and videos.

Deployment architecture for compute intensive applications

Deployment architecture for compute intensive applications such as Data Analytics, Media Transcoding, etc. In this we have web, application, storage, computing/analytics and database tiers. Data analysis jobs (such as MapReduce) jobs are submitted to the analytics tier from the application servers. The jobs are queued for execution and upon completion the analyzed data is presented from the application servers.

Cloud Application Design Methodologies

In this we discuss about several design methodologies for cloud computing

Service Oriented Architecture (SOA)

Service Oriented Architecture (SOA) is a well established architectural approach for designing and developing applications in the form services that can be shared and reused. SOA services are developed as loosely coupled modules with no hard-wired calls embedded in the services. The services communicate with each other by passing messages. The layers of Service Oriented Architecture (SOA) is

Business Systems: This layer consists of custom built applications and legacy systems such as Enterprise resource planning(ERP),Customer Relationship management(CRM),Supply chain management(SCM).

Service Components: The service components allow the layers above to interact with the business systems.

Composite Services: These are coarse-grained services which are composed of two or more service components.

Orchestrated Business Processes: Composite services can be orchestrated to create higher level business processes. In this layers the compositions and orchestrations of the composite services are defined to create business processes.

Presentation Services: This is the topmost layer that includes user interfaces that exposes the services and the orchestrated business processes to the users.

Enterprise Service Bus: This layer integrates the services through adapters, routing, transformation and messaging mechanisms.

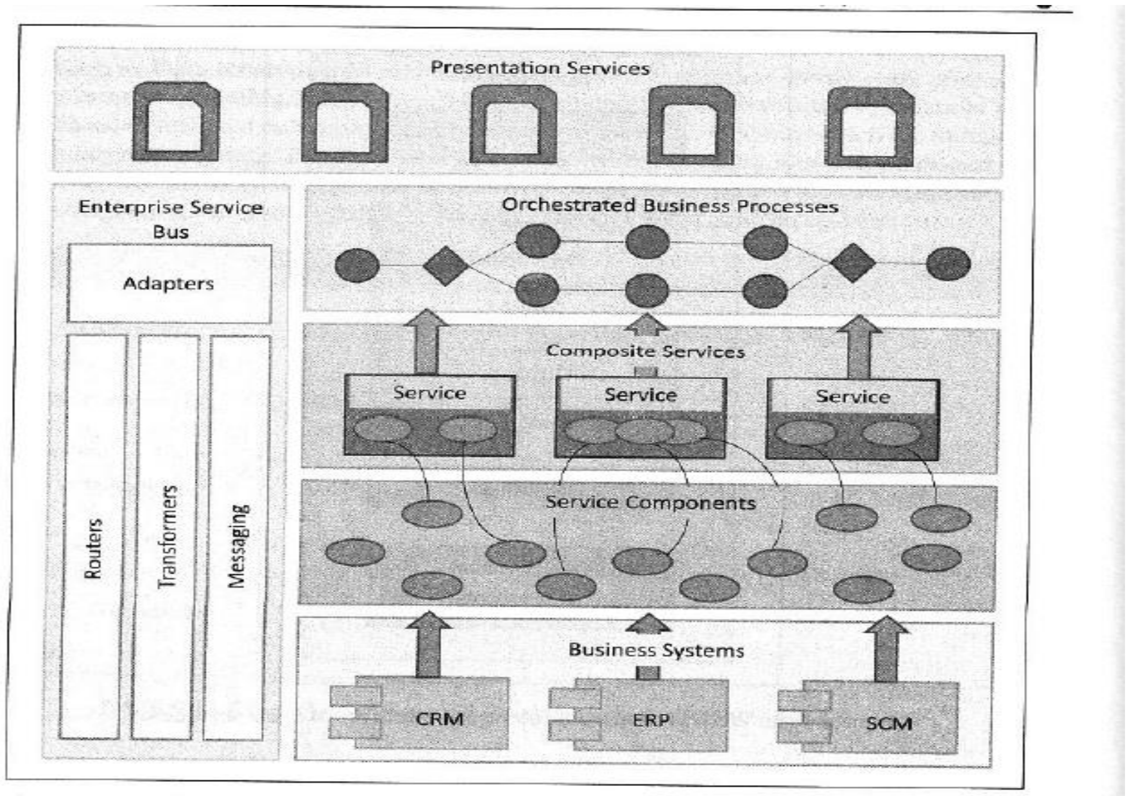
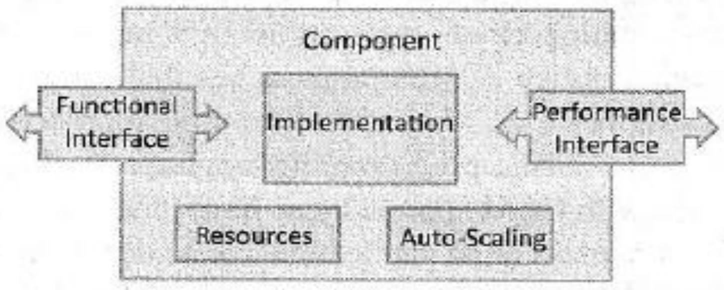


Figure 5.5: Layers of service oriented architecture

Cloud Component Model

Cloud Component Model is an application design methodology that provides a flexible way of creating cloud applications in a rapid, convenient and platform independent manner. Cloud applications designed with CCM approach can have innovative hybrid deployments in which different components of an application can be deployed on cloud infrastructure and platforms of different cloud vendors. This figure shows the architecture of a CCM component.

Each component takes specific inputs, performs a pre-defined set of actions and produces the desired outputs. Components offer their functions as services through a functional interface which can be used by other components. Components report their performance to a performance database through a performance interface. Components have number of resources such as web pages, images, documents, database tables, etc. Auto-scaling performance constraints and conditions can be specified for each component.



(b)

The Architecture Design has following properties

Loose Coupling: Components in the cloud component model are loosely coupled.

Asynchronous Communication: Loosely coupled components communicate asynchronously through message based communication.

Stateless Design: CCM are stateless. By storing session state outside of the component.

<u>REST architecture</u>

The REST architectural constraints are as follows:

Client-Server: The principle behind the client-server constraint is the separation of concerns. For example, clients should not be concerned with the storage of data which is a concern of the server.

Stateless: Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server. The session state is kept entirely on the client.

Cacheable: If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.

Layered System: Layered system constraint constrains the behavior of components such that each component cannot see beyond the immediate layer with which they are interacting.

Uniform Interface: Uniform Interface constraint requires that the method of communication between a client and a server must be uniform.