Northeastern University – Silicon Valley

CS 6620 Cloud Computing

Final Exam [250 points]

This is an open book exam. Scan answers to PDF. Duration: 1 day take home.

Answer all questions, with brief but complete explanations. K*eep your answers concise and to the point. Please write legibly. Write your name on Page 1. Please use Diagrams to explain design. Avoid copy-paste. Do not refer to the Web resources, refer only to class notes and textbook. Please do not copy and paste from *ANY* resource including the Web, class notes etc. This will disqualify the exam.*

This is an open-ended exam. There is no one correct answer.

The exam should be answered using your own knowledge and study.

I will not be answering any questions during the exam day.

Please do not start discussions on Piazza or any forum with other students or the TAs or the professor.

Use the resources provided, your class notes and texts and complete your answers by yourself.

Instructions

Consider the two different practical design web applications provided as PDF for this exam (Yelp and Dropbox). Study the given resources(PDF) and gain a good understanding first.

Part 1: Basics

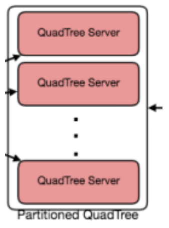
a. Is Yelp hosted as SaaS only, or is it an IaaS? Why? Do you need multi-tenancy for Yelp? Why or why not? Please explain. [5 points]

SaaS. According to the description of Yelp, their service is storing information about different places so that users can perform a search on them. Upon querying, the service will return a list of places around the user. Apparently, the service is to support search function on software. Thus, Yelp is hosted as SaaS.

Yelp need multi-tenancy. Multitenancy is a reference to the mode of operation of software where multiple independent instances of one or multiple applications operate in a shared environment. The instances (tenants) are logically isolated, but physically integrated.

According to the definition of multi-tenancy, for the reason why Yelp need multi-tenancy, there are two reasons:

1. Putting all the data in one big database machine was a lot more expensive than several smaller machines.
2. It wasn't really safe to put all these uses on the same physical machine because of performance isolation and security considerations.



b. Dropbox or Google Drive: Is this platform hosted as SaaS only, or is it an IaaS or both? Why? Do you need multi-tenancy for Dropbox? Why or why not? Please explain. [5 points]

The Dropbox platform is hosted as both IaaS and SaaS. According to the description of Dropbox, their service - Cloud file storage enables users to store their data on remote servers. Usually, these servers are maintained by cloud storage providers and made available to users over a network (typically through the Internet). So they have both software and infrastructure as a service.

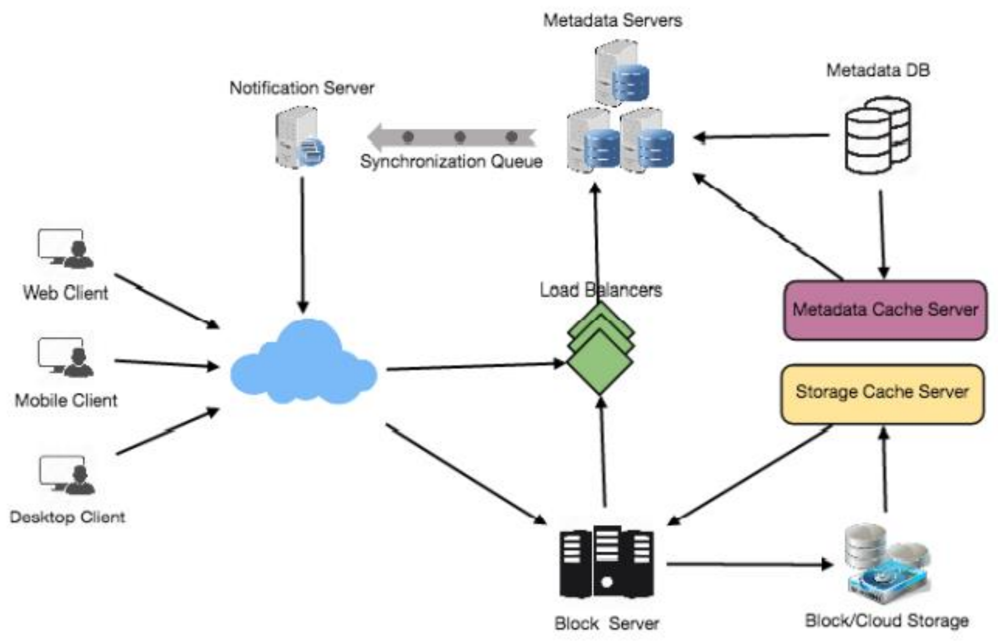
Yes, Dropbox need multi-tenancy. Multitenancy is a reference to the mode of operation of software where multiple independent instances of one or multiple applications operate in a shared environment. The instances (tenants) are logically isolated, but physically integrated.

According to the definition of multi-tenancy, for the reason why Dropbox need multi-tenancy, there are two reasons:

1. Putting all the data in one big database machine was a lot more expensive than several smaller machines.

2. It wasn't really safe to put all these uses on the same physical machine because of performance isolation and security considerations.

The following figure shows the detailed component design for Dropbox.



(Please focus only on the **Dropbox** platform for the next two parts)

Part 2: Usage and Scaling Calculations

c. **User and functionality Calculations**: For the Dropbox platform only, develop detailed user/usage calculations on the expected total users and concurrent users (CCU), making reasonable assumptions and following this reference: https://cloudkul.com/blog/what-is-concurrent-users/ make all reasonable assumptions about the data usage patterns of users, their geographical locations etc. [20 points]

For the platform Dropbox, concurrent users are a standard way of planning, measuring, and managing service capacity. To assess the service capability, it is fair to look at the peak concurrent users for a time period and the estimation of concurrent users will be used to assess whether your system and services are capable of handling the peak traffic on the website or not.

Basic formula for computing concurrent users is as follows:

*Concurrent visitors = Per Day visits / Peak hours * (60/Average duration per visit in minutes)*

Reasonable assumptions:

1. Access to the Dropbox platform is portable by multiple devices from various geographical locations at any time.
2. Around 1000 users visit the website daily.
3. Approximately 30 minutes were taken by a single user on the website.
4. The peak hours or high traffic is from 2:00 P.M to 6:00 P.M (i.e, 4 hours).

Concurrent visitors = 1000 / 4 * (60/30) = 1000 / 4*2 = 1000 /8 = 125

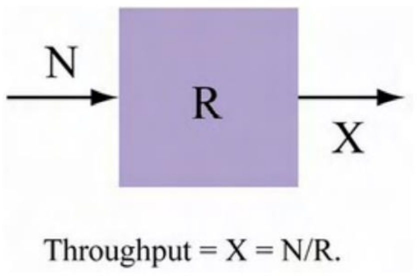
As a result, the approximate number of concurrent users on Dropbox platform is 125.

d. **Scalability Calculations:** For the Dropbox platform only, develop detailed Scalability calculations on the expected **growth** in the total users and concurrent users (CCU), making reasonable assumptions and following this reference: Design for growth up to 1,000,000 users https://www.simform.com/blog/building-scalable-application-aws-platform/ make reasonable assumptions about growth patterns of users, their geographical locations etc. [30 points]

Informally, engineers describe the challenge of dealing with large audiences and high demand as a
problem of scalability. More specifically, a Web application can scale if it continues to be available
and functional at consistent speeds as the number of users and requests continues to grow, even to
very high numbers.

There are three techniques can be employed to achieve scalability
• Increase the resources (bigger machines, more disk, more memory).
• Improve the software.
• Increase the resources and improve the software

Scalability calculations
Little's law: if this box contains an average of N users, and the average user spends R seconds in
that box, then the throughput X of that box is roughly:



Reasonable assumptions:

1. A simple architecture with reduced operational costs is needed.
2. The requirement is a system that can scale without consuming much power.
3. The manager need an easy to install and scalable system with lower licensing costs.
4. The compatibility of applications maintained should be considered.
5. The number of concurrent Users(N) is 1,000,000.
6. Average Response Time(R) is 500 ms (0.5s)

Result:

For this assumption, vertical scaling would be chosen.

Also, using Service Oriented Architecture(SOA) for better flexibility while designing large scale web applications.

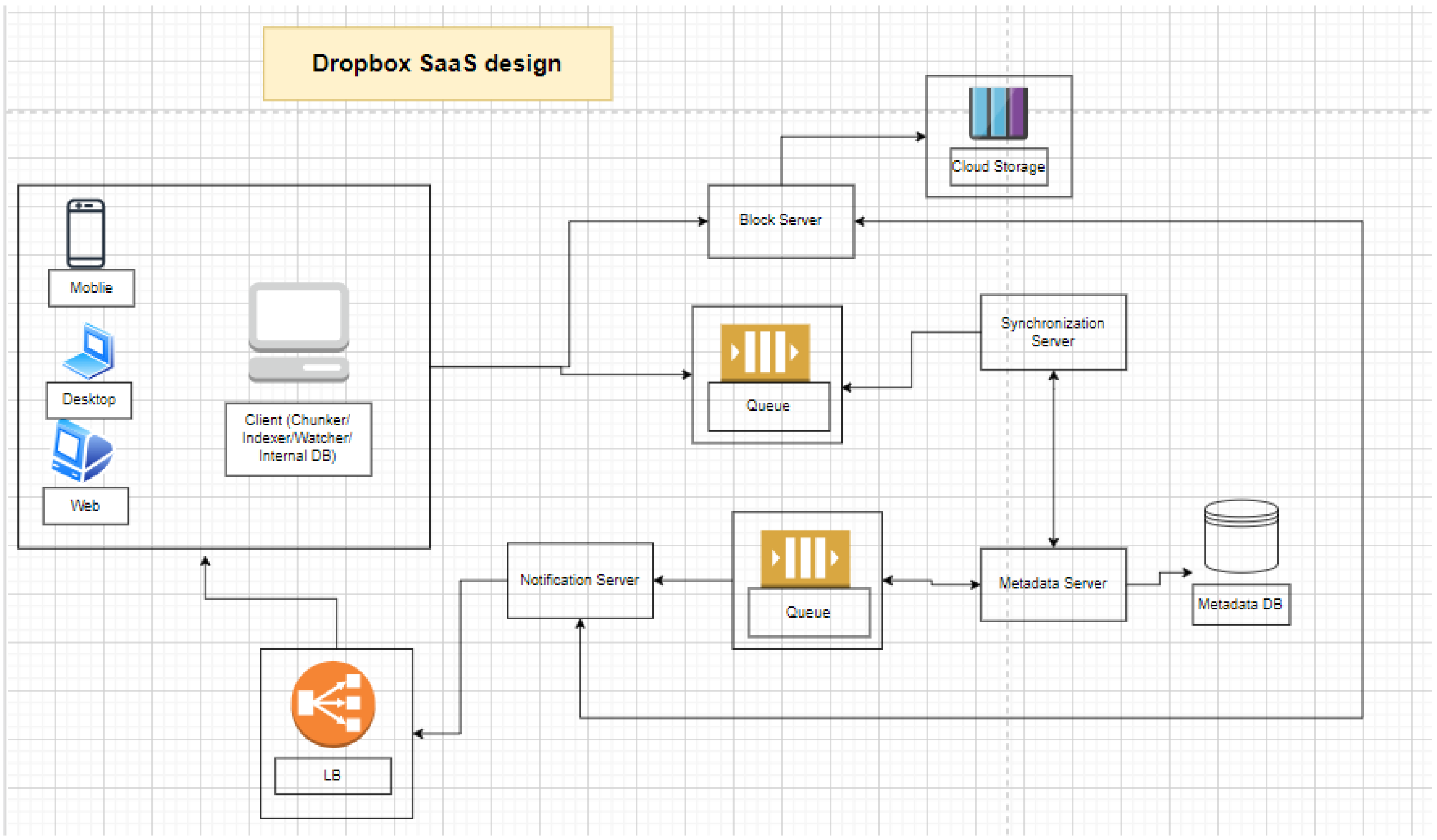
Throughput = N/R = 1000000/0.5 = 2000000 tps

Part 3: Detailed Design and Mapping to AWS and Open stack

Note: Your design should take into consideration CCU (i. e. concurrent users), scalability and growth calculations from Part 2 above.

e. **Design IaaS and SaaS separately** for this Dropbox platform. This means - Provide a basic architectural diagram for this web app, identifying all key business components you need to provide for SaaS. Similarly, provide the detailed Infrastructure design (VMs and Network level) for IaaS. If multitenancy is relevant, explain how you will achieve this in your design. [100 points]

SaaS:



Key business components:

Load balancer

Block Server

Cloud Storage

Queue

Synchronization Server

Metadata Server

Metadata DB

Notification Server

API Service

IaaS Design:

Create VPC

Setup the Internet Gateway

Attach the VPC to the internet gateway

Create Subnets

Create Route Tables

Route the traffic to the internet through the internet gateway

Create the NAT Gateway

Edit the private route table to make use of the NAT gateway to access the internet

Create Elastic Load Balancer

As multitenancy is relevant to Dropbox, I would achieve this in your design by set different server and service to different tenant.

f. **Hosting Dropbox on AWS**: For the Dropbox platform only, provide a detailed mapping of the various components you need on AWS to be able to build this application and host it on AWS. Identify and detail the mapping of at least 10 key AWS components needed, with a brief explanation of why they are needed and what they provide for your platform. [40 points]

1. EC2

For the Dropbox platform hosting tasks, Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud.

2. ECS

For the Dropbox platform hosting tasks, Amazon ECS is a fully managed container orchestration service that makes it easy for developers to deploy, manage, and scale containerized applications.

3. RDS

For the Dropbox platform hosting tasks, Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud.

4. IAM

For the Dropbox platform hosting tasks, AWS Identity and Access Management (IAM) provides fine-grained access control across all of AWS. With IAM, users can specify who can access which services and resources, and under which conditions.

5. VPC

For the Dropbox platform hosting tasks, Amazon Virtual Private Cloud (Amazon VPC) gives users full control over your virtual networking environment, including resource placement, connectivity, and security.

6. Lambda

For the Dropbox platform hosting tasks, AWS Lambda is a serverless, event-driven compute service that lets users run code for virtually any type of application or backend service without provisioning or managing servers.

7. S3

For the Dropbox platform hosting tasks, Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

8. Load Balancer

For the Dropbox platform hosting tasks, a load balancer is a device that acts as a reverse proxy and distributes network or application traffic across a number of servers.

9. CodeCommit/CodeBuild/CodeDeploy/CodePipeline

For the Dropbox platform hosting tasks, these four services could be helpful for DevOps, adopting the culture of continuous integration and continuous delivery/deployment (CI/CD), where a commit or change to code passes through various automated stage gates

10. Elasticsearch

For the Dropbox platform hosting tasks, Elasticsearch is a new open source, distributed search and log analytics suite derived, which could be useful for interactive log analytics, real-time application monitoring, and website search.

g. **Hosting Dropbox on Open stack:** For the Dropbox platform only, provide a detailed mapping of the various components you need on AWS to be able to build this application and host it on Open stack. Identify and detail the mapping of at least 10 key Open stack components needed, with a brief explanation of why they are needed and what they provide for your platform. [40 points]

1. Compute (Nova)

For the Dropbox platform hosting tasks, OpenStack Compute is a cloud computing fabric controller, which manages pools of computer resources and work with virtualization technologies, bare metals, and high-performance computing configurations. Nova's architecture provides flexibility to design the cloud with no proprietary software or hardware requirements and also delivers the ability to integrate the legacy systems and third-party products.

2. API Proxies (EC2API)

For the Dropbox platform hosting tasks, EC2API is a compatibility layer for Amazon EC2 API service in OpenStack. It replaces the previously built-in nova EC2-API service.

2. Image Service (Glance)

For the Dropbox platform hosting tasks, OpenStack image service offers discovering, registering, and restoring virtual machine images. Glance has client-server architecture and delivers a user REST API, which allows querying of virtual machine image metadata and also retrieval of the actual image. While deploying new virtual machine instances, Glance uses the stored images as templates.

3. Object Storage (Swift)

For the Dropbox platform hosting tasks, OpenStack Swift creates redundant, scalable data storage to store petabytes of accessible data. The stored data can be leveraged, retrieved and updated. It has a distributed architecture, providing greater redundancy, scalability, and performance, with no central point of control.

4. Dashboard (Horizon)

For the Dropbox platform hosting tasks, Horizon is the authorized implementation of OpenStack's Dashboard, which is the only graphical interface to automate cloud-based resources. To service providers and other commercial vendors, it supports with third party services such as monitoring, billing, and other management tools. Developers can automate tools to manage OpenStack resources using EC2 compatibility API or the native OpenStack API.

5. Identity Service (Keystone)

For the Dropbox platform hosting tasks, Keystone provides a central list of users, mapped against all the OpenStack services, which they can access. It integrates with existing backend services such as LDAP while acting as a common authentication system across the cloud computing system.

Keystone supports various forms of authentication like standard username & password credentials, AWS-style (Amazon Web Services) logins and token-based systems. Additionally, the catalog provides an endpoint registry with a queryable list of the services deployed in an OpenStack cloud.

6. Networking (Neutron)

For the Dropbox platform hosting tasks, Neutron provides networking capability like managing networks and IP addresses for OpenStack. It ensures that the network is not a limiting factor in a cloud deployment and offers users with self-service ability over network configurations. OpenStack networking allows users to create their own networks and connect devices and servers to one or more networks. Developers can use SDN technology to support great levels of multi-tenancy and massive scale.

Neutron also offers an extension framework, which supports deploying and managing of other network services such as virtual private networks (VPN), firewalls, load balancing, and intrusion detection system (IDS)

7. Block Storage (Cinder)

For the Dropbox platform hosting tasks, OpenStack Cinder delivers determined block-level storage devices for application with OpenStack compute instances. A cloud user can manage their storage needs by integrating block storage volumes with Dashboard and Nova.

Cinder can use storage platforms such as Linux server, EMC (ScaleIO, VMAX, and VNX), Ceph, Coraid, CloudByte, IBM, Hitachi data systems, SAN volume controller, etc. It is appropriate for expandable file systems and database storage.

8. Telemetry (Ceilometer)

For the Dropbox platform hosting tasks, Ceilometer delivers a single point of contact for billing systems obtaining all of the measurements to authorize customer billing across all OpenStack core components. By monitoring notifications from existing services, developers can collect the data and may configure the type of data to meet their operating requirements.

9. Orchestration (Heat)

For the Dropbox platform hosting tasks, Heat is a service to orchestrate multiple composite cloud applications through both the CloudFormation-compatible Query API and OpenStack-native REST API, using the AWS CloudFormation template format.

h. Provide **conclusions and final design recommendation**. which of the two cloud platforms will you use for hosting Dropbox? Please justify your answer. [10 points]

OpenStack and AWS are undoubtedly some of the most popular cloud technologies in both public and private cloud space.

But there are still some differences between them.
Apparently, as the Dropbox would serve more than1000000 concurrent users, which is quite a lot of users, the cost of AWS would be much greater than the Open stack. So I would mainly use Open Stack for hosting.

For the greater suggestion of Dropbox, combine the two cloud platforms could be a more popular choice as many big businesses such as Netflix and YouTube. By use both at the same time in a hybrid/multi-cloud architecture, this approach enables to optimize infrastructure costs even further. Start small with AWS and build a cost-effective private cloud platform based on OpenStack once the demand for resources increases. Then move the majority of the workloads there, benefitting from cost reduction, while still using highly scalable AWS resources during heavy load periods and for periodic compute-intensive operations execution.