Cloud Summary

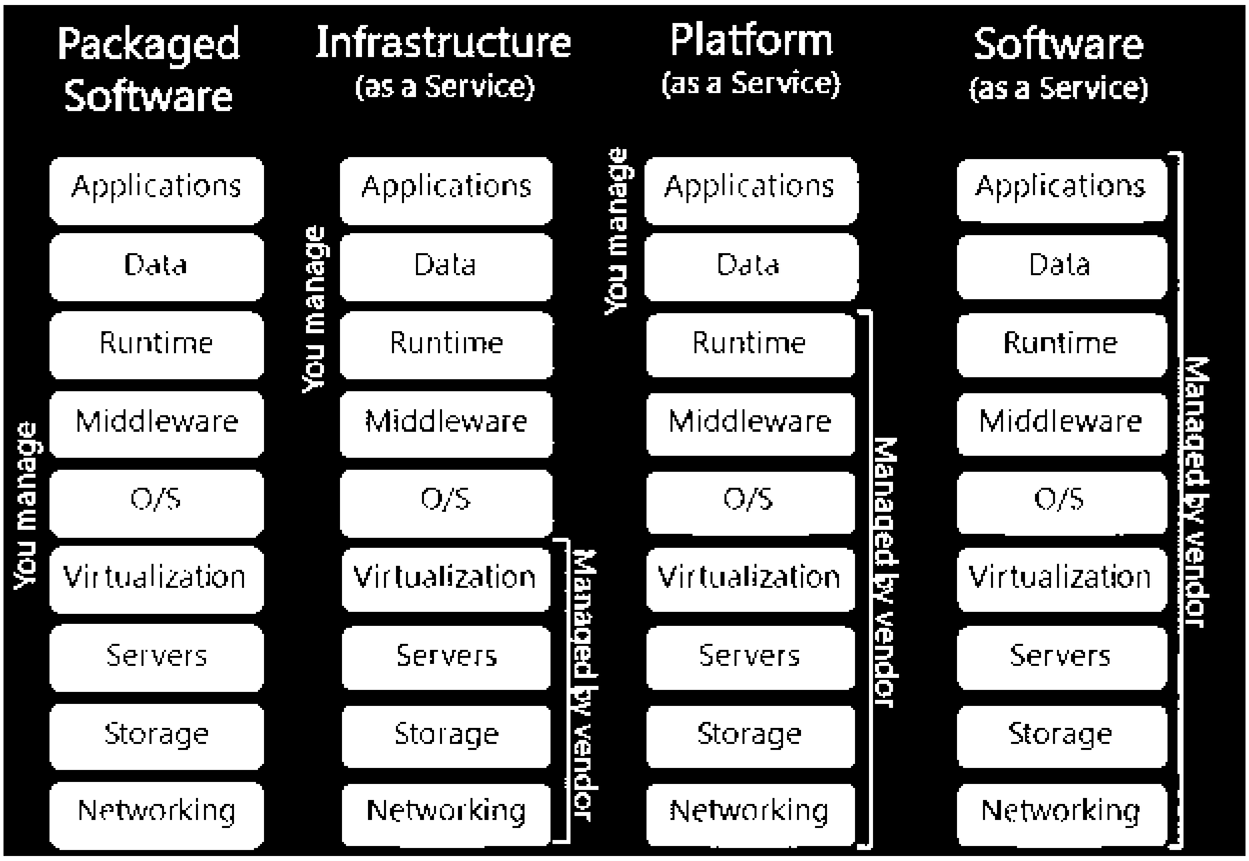
Lecture 1:

- Cloud computing is the **delivery of computing as a service** rather than a product, whereby shared **resources**, **software**, and **information** are provided to computers and other devices as a metered service over a network (typically the Internet).
- If we manage our data ourselves, it's personal but time consuming.
- A cloud is a data center hardware and software that the vendors use to offer the computing resources.
- The promise of the cloud is the transformation of IT from a product to a service.
- Why cloud computing?
    - Pay-as-you-go economic model:
        - Reduce capital expenditure.
        - No upfront cost.
        - Reduce time to market.
    - Simplified IT management:
        - It is the providers responsibility to manage the details.
    - Scale quickly and effortlessly:
        - Resources can be rented and released as required.
    - Flexible options:
        - Access from any machine connected to the internet.
    - Resource utilization:
        - Better utilization of CPU, storage, and bandwidth.
    - Carbon footprint decreased
        - Sharing of resources means less servers, less power, and less emissions.

Lecture 2:

- Intended Audience:
    - Students
    - Researchers and professors
    - Cloud service providers
    - Cloud service consumers
- NIST Definition of Cloud Computing:
    - Cloud computing is a model for enabling **ubiquitous**, convenient, **on demand** network access to a **shared** pool of **configurable** computing resources that can be **rapidly provisioned** and released with **minimal management** effort or service provider interaction.

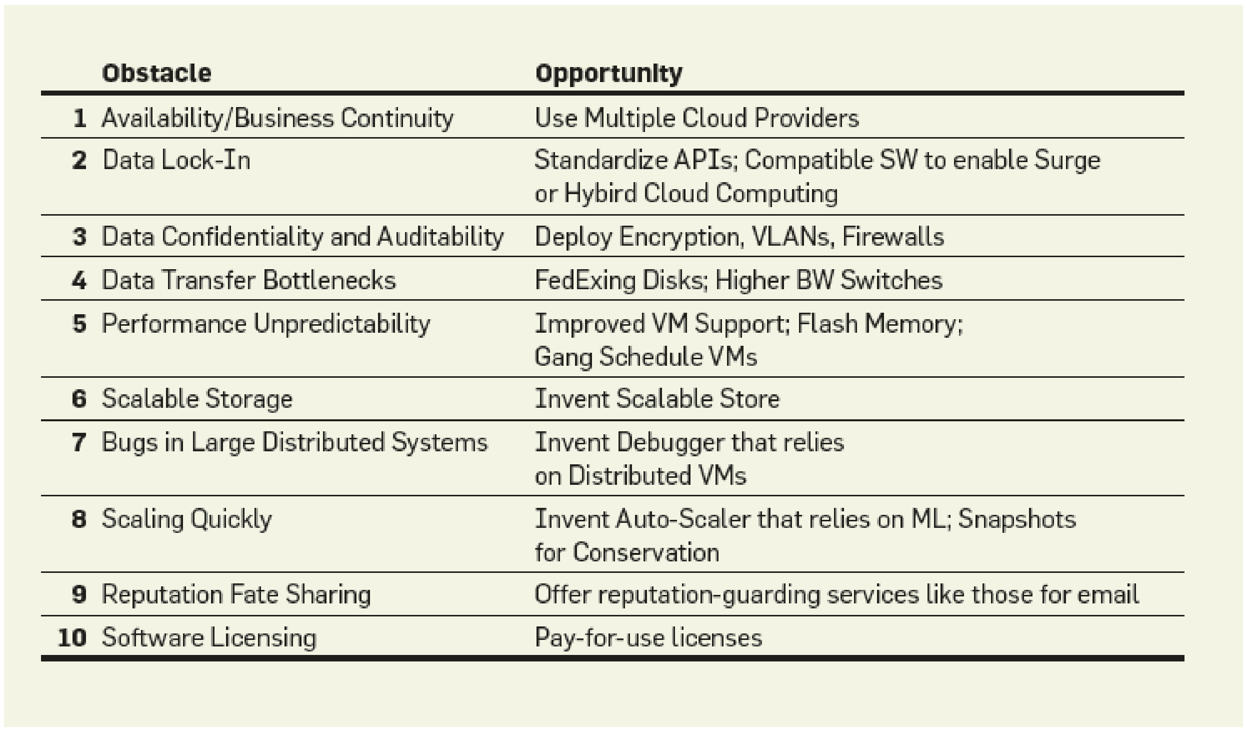
- Five Cloud Computing Characteristics:
  - On demand self-service.
  - Broad network access.
  - Resource pooling.
  - Rapid elasticity.
  - Measured service.
- Three Cloud Service Models:
  - Software-as-a-service (SaaS) [applications]
    - Finished applications that you rent and customize.
  - Platform-as-a-service (PaaS) [frameworks] (developer interest)
    - Developer platform that drives developer productivity.
  - Infrastructure-as-a service (IaaS) [hardware]
    - Deployment platform that abstracts the infrastructure.



- Four Cloud Deployment Models:
  - Public cloud
    - Infrastructure is provisioned for open use by **general public**.
  - Private cloud
    - Infrastructure is provisioned for exclusive use by a **single organization**.
  - Community cloud
    - Infrastructure is provisioned for exclusive use by a specific community of organizations that have shared concerns.
  - Hybrid cloud

Lecture 3:

- **Cloud Consumer:** A person or organization that maintains a business relationship with, and uses service from, cloud providers.
- **Cloud Provider:** A person, organization, or entity responsible for making a service available to interested parties.
- **Cloud Auditor:** A party that can conduct independent assessment of cloud services, information system operations, performance, and security of the cloud implementation.
- **Cloud Carrier:** An intermediary that provides connectivity and transport of cloud services from cloud providers to cloud consumers.
- **Cloud Broker:** An entity that manages the use, performance, and delivery of cloud services, and negotiates relationships between cloud providers and cloud consumers.
- SLA = service-level agreement.
- Service layer is an interface to use the service itself.
- Cloud Provider Major Activities:
    - Service deployment
    - Service orchestration
    - Cloud service management
        - Business support
        - Provisioning and configuration
        - Portability and interoperability (interaction between installed platform through interface)
    - Security
    - Privacy
- Cloud obstacles and opportunities



	Obstacle	Opportunity
1	Availability/Business Continuity	Use Multiple Cloud Providers
2	Data Lock-In	Standardize APIs; Compatible SW to enable Surge or Hybird Cloud Computing
3	Data Confidentiality and Auditability	Deploy Encryption, VLANs, Firewalls
4	Data Transfer Bottlenecks	FedExing Disks; Higher BW Switches
5	Performance Unpredictability	Improved VM Support; Flash Memory; Gang Schedule VMs
6	Scalable Storage	Invent Scalable Store
7	Bugs in Large Distributed Systems	Invent Debugger that relies on Distributed VMs
8	Scaling Quickly	Invent Auto-Scaler that relies on ML; Snapshots for Conservation
9	Reputation Fate Sharing	Offer reputation-guarding services like those for email
10	Software Licensing	Pay-for-use licenses

Lecture 4:

- **Virtualization:**
  - It is a technology to run **multiple** same or different OSs on a single physical system which are completely isolated from each other.
  - It is defined as the **abstraction** over computing resources into multiple virtual machines (VMs).
  - Multiple VMs can run on a single physical system.
  - A VM provides interface identical to underlying **bare hardware**.
  - It is the process by which one computer behaves as many computers.
- Virtualization used to improve **IT throughput** and **costs** by using physical resources as a pool from which virtual resources can be allocated.
- **Dual Boot System:** A computer system in which two operating systems are installed on the same hard drive.
- **Virtualization System:** A system pretends to be two or more of the same system.
- VMs can be scaled **up** and **down** on demand with a high level of resources' abstraction.
- Virtualization provides on demand **cloning** and **live migration** services which improve **reliability**.
- VIM: Virtualization Infrastructure Management.
- Several components of virtualization:
  - **Host** - underlying hardware system.
  - **Virtual Machine Manager (VMM)** or **hypervisor** - creates and runs virtual machines by providing interface that is **identical** to the host.
  - **Guest** - process provided with virtual copy of the host, usually an OS.
- **Virtualization Layer:** middleware between the underlying hardware and virtual machines represented in the system, also known as **Virtual Machine Monitor (VMM)** or **hypervisor**.

Server without Virtualization	**Server with Virtualization**
Single OS can run at a time within a server.	Can run multiple Oss simultaneously.
Running multiple applications on same machine creates conflict.	Each virtual machine is independent and can be provisioned any time.
Underutilization of resources.	Efficient utilization of hardware resources.
Inflexible and costly infrastructure.	Save electricity, initial cost to buy servers, space.
Hardware changes require manual effort and access to the physical server.	Easy to manage and monitor virtual machines centrally.

- **Hypervisor:** A software that allows multiple OSs (guest) to share a single hardware host.
- However, the hypervisor is actually
  - **Controlling** the host processor and resources.
  - **Allocating** what is needed to each operating system in turn.
  - **Making sure** that the guest operating systems (called Virtual Machines(VMs)) cannot disrupt each other.
- The term hypervisor is a variant of **supervisor**, a traditional term for the kernel of an operating system.
- Hypervisor Types:
  - Full virtualization:
    - It is called native or bare metal hypervisors.
    - It directly sitting on top of the bare hardware devices.
    - Hypervisors enable to run multi unmodified guest operating system.
    - No host OS is used here.
    - Improved reliability, security, and productivity.
  - Para Virtualization:
    - These hypervisors run on a conventional operating system (host OS) just as other computer programs do.
    - A guest OS runs as a process on the host OS.
    - Improves performance.
    - Lower overhead.
- There are some challenges in full virtualization:
  - Security issues.
  - Simulation of privileged operations.
  - The effects of every operation performed within a given virtual machine must be kept within that virtual machine.
  - Some machine instructions of guest virtual machine can be executed directly by the hardware.
  - Some instructions of guest virtual machine cannot be allowed to execute directly; instead, they must be trapped and simulated.
  - Some hardware is not easy to be used for full virtualization.
- Benefits of virtualization:
  - Consolidation.
  - Sharing of resources.
  - Isolation.
  - Encapsulation.
  - Hardware Independence.
  - Portability.

- Virtualization in cloud computing:
    - You don t need to own hardware.
    - Resources are rented as needed from a cloud.
    - Various providers allow creating virtual servers.
    - You get billed only for what you used.
    - Benefits:
        - Reduce capital expenses.
        - Reduce maintenance and operation expenses.
        - Reduce physical space needed in data centers.
        - Resource management, migration, maintainability, high availability, and fault tolerance are other benefits.
- **Containers** are an abstraction at the app layer that packages code and dependencies together.
    - Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space.

Lecture 5:

- Virtualization: Creation of a virtual version of hardware using software (hypervisor).
- By running multiple virtual machines simultaneously, a physical server can be utilized efficiently.
- Virtualization Evolution:
    - Capital Savings
        - Small scale consolidation.
        - Saves servers/network management/maintenance engineers.
    - Operation and Management Savings
        - [Better] production consolidation.
        - Business continuity.
        - Workload balancing.
    - SOA (Service oriented architecture) Architecture
        - Automate IT processes.
        - Create resource pools.
- Advantages of virtualization:
    - Zero downtime maintenance.
    - Instant provisioning.
    - Business continuity, backups, and automated restoration.
    - Virtual hardware supports legacy operating systems efficiently.
    - Freedom from vendor-imposed upgrade cycles.
    - Pooling hardware resource.
    - Dynamic resource sharing.
    - Security and fault isolation.
- Impact of Virtualization:
    - Hard cost savings.
    - Operational efficiency.
- Cloud Computing offers infrastructure as a service which is based on:
    - Pay as you use model.
    - On demand computing model.
- To provide infrastructure as service, the provisioning of the cloud infrastructure in data centers is a prerequisite which is time consuming.
- There are two core services enable the users to get the best out of the IaaS model:
    - Virtual machine provisioning.
    - Migration services.
- By emerging of virtualization technology and the cloud computing IaaS model, it is just a matter of minutes to achieve the same task.
- It is an expensive operation to maintain or upgrade a main server that has lots of applications and users, with the advance of migration services, these tasks are very easy and need no time to accomplish.

- The main issue about the virtualization layer is that it **schedules and allocates** the physical resource, and makes each virtual machine think that it totally owns the whole underlying hardware's physical resource.
- Virtualization at Instruction Set Architecture (ISA) Level:
  - Advantages:
    - It can run a large amount of legacy binary codes.
    - Best application **flexibility**.
  - Disadvantages:
    - It may require tens or hundreds of native target ISA instructions to perform one function, which is relatively **slow**.
- Virtualization at Hardware Abstraction Level:
  - Advantages:
    - Has higher performance and good application isolation.
  - Disadvantage:
    - Very expensive to implement (complexity).
- Virtualization at OS Level:
  - It is an abstraction layer between **traditional OS** and **user applications**.
  - This virtualization creates isolated **containers** on a single physical server.
  - Advantages:
    - Minimum startup/shutdown costs.
    - Can easily synchronize with its environment.
  - Disadvantages:
    - All VMs at the OS level must have the same kind of Host OS.
    - Poor application flexibility and isolation.
- Virtualization at User Application Level:
  - This layer sits as an application program on top of an operating system.
  - Advantages:
    - It has the best application isolation.
    - Support code portability.
  - Disadvantages:
    - Low performance, low application flexibility and high implementation complexity.
- Full Virtualization: Sharing the resources of a single hardware across multiple environments.
- Para Virtualization: **Host OS** provides an abstraction layer for running virtual **guest OS.**
- OS Level Virtualization: It is the enabling technology and creates virtual machines that allows a single machine to act as if it were many machines.

- Different Virtualization Types in Cloud Computing:
    - Hardware Virtualization:
        - Hardware virtualization is also known as hardware assisted virtualization or server virtualization.
        - Increased processing power as a result of maximized hardware utilization and application uptime.
    - Software Virtualization:
        - It involves the creation of an operation of multiple virtual environments on the host machine.
    - Memory Virtualization:
        - Physical memory across different servers is aggregated into a single virtualized memory pool.
        - Subtypes:
            - Application-level-control.
            - Operating system-level-control.
    - Storage Virtualization:
        - Multiple physical storage devices are grouped together, which then appear as a single storage device.
        - Reduced downtime, load balancing.
        - Better optimization of performance and speed.
        - Subtypes:
            - Block virtualization.
            - File virtualization.
    - Data Virtualization.
    - Network Virtualization:
        - It enables restriction of file movement across networks and enhances **security** and allows better monitoring and identification of data usage which lets the network administrators **scale up** the network appropriately. It also increases **reliability** as a disruption in one network doesn’t affect other networks, and the diagnosis is easier.
    - Desktop Virtualization:
        - The user s desktop is stored on a remote server, allowing the user to access his desktop from any device or location.

Big Data lecture:

- Cloud Computing
  - Large Internet services running [Google & Facebook].
  - External customers rent cycles.
  - Features:
    - Scalability.
    - Pay for what you use.
    - Ease of use.
- ERP → CRM → WEB → BIG DATA
  Increase in data size, variety, complexity.
- Big Data = transactions + interactions + observations.
- Big Data: 3 V's
  - Volume
  - Variety
  - Velocity
- Big data is the term for a collection of data sets so large and complex that it becomes difficult to process using on hand database management tools or traditional data processing applications.
- Goals of HDFS:
  - Very Large Distributed File System.
  - Files are replicated to handle hardware failure.
  - Detect failures and recovers from them.
  - Optimized for Batch Processing.
  - User Space runs on heterogeneous OS.
- Design of HDFS:
  - Single Namespace for entire cluster.
  - Data Coherency.
  - Files are broken up into blocks.
  - Intelligent client.
- Functions of a NameNode:
  - Manages File System Namespace.
  - Cluster Configuration Management.
  - Replication Engine for Blocks.
  - To ensure high availability.
- NameNode Metadata
  - Metadata in Memory.
  - Types of metadata.
  - A Transaction Log.

- **Advantages of the 3x replication schema in HDFS:**
    - **Supports parallel processing.**
    - **Faster data analysis.**
    - **Ensures fault tolerance.**
    - **Manages cluster resources.**
- MapReduce:
    - Data parallel programming model for clusters of commodity machines.
    - Is a programming model for efficient distributed computing
    - MapReduce goals: Scalability and cost-efficiency.
    - A good fit for a lot of applications:
        - Log processing.
        - Web index building.
    - Features:
        - Fine grained Map and Reduce tasks.
        - Automatic re-execution on failure.
        - Locality optimizations.
- Challenges of Cloud Environment:
    - Cheap nodes fail
        - Solution: Build fault tolerance into system
    - Low bandwidth
        - Solution: Push computation to the data
    - Programming distributed systems is hard
        - Solution: users write data parallel map and reduce functions.
- Advantages of Hadoop:
    - Vast amounts of data
    - Economic
    - Efficient
    - Scalable
    - Reliable
- Applications not for Hadoop:
    - Low latency data access.
    - Lots of small files.
    - Multiple writers, arbitrary file modifications.
- Hadoop components:
    - Distributed file system (HDFS):
        - Single namespace for entire cluster.
        - Replicates data 3x for fault tolerance.
    - MapReduce
    - Yarn

- YARN is the prerequisite for Enterprise Hadoop.
- In a YARN cluster, there are two types of hosts:
  - The ResourceManager is the master daemon that communicates with the client.
  - A NodeManager is a worker daemon that tracks its own local resources and communicates its resource configuration to the ResourceManager.
- Containers: a request to hold resources on the YARN cluster.
- Yarn application: It is a YARN client program that is made up of one or more tasks.
- ApplicationMaster:
  - It helps coordinate tasks on the YARN cluster for each running application
  - It is the first process run after the application starts.

DevOps lecture:

- DevOps definition:
    - It is the practice of **operations** and **development** engineers participating together in the entire service lifecycle.
- What made us think of DevOps?
    - Delivering a service from beginning to end to the users is too **slow** and **error-prone**, and every delay is a loss.
- Symptoms that you need DevOps:
    - Defects released into production/failed releases.
    - Inability to diagnose production issues.
    - Problems appear in some environments only.
    - Long response delays from other teams.
- Why does this problem exist?
    - The business has demanded the wrong things out of IT, which are
        - Cost sensitive
        - Risk averse
- Why there are gaps between the development team and operation team?
    - Dev view:
        - Dev systems may not be the same as production system.
        - Developers will have faster turnaround time.
        - Not very much concerned about deployment impact.
    - Ops view:
        - Rewarded mainly for uptime.
        - Less turnaround time.
        - Very much concerned about deployment impact.
- Dev and ops:
    - Operations staff use many of the same techniques as developers for their system work.
    - Closely monitors the Dev -Test – Prod pipeline for each deployment with immediate feedback.
- Aims of DevOps:
    - Shorten the system's development life cycle.
    - Provide continuous delivery of high software quality.
    - Reduces risk and cost.
- DevOps principles [CAMS]:
    - Culture.
    - Automation.
    - Measurement.
    - Sharing.

- DevOps practices:
    - Automated testing.
    - Proactive monitoring.
    - Kanban/Scrum.
    - Configuration management.
    - CI/CD.
    - Continuous Integration/testing/delivery.
    - Virtualization/cloud/containers.
- How are DevOps and Cloud related?
    - With cloud, there's no limit on how much functionality a company can access.
    - In the same way, DevOps supports an agile environment for everyone involved. Both solutions have their own benefits to consider for speed and productivity.
- DevOps as a service collects the development tools into a **single toolset** that is hosted in the cloud. The objective is to ensure that developers use a common toolset and that every action is tracked to promote continuous delivery, continuous integration, and business value.
- Pros of DevOps as a service:
    - Easier to collaborate.
    - Faster testing and deployment.
    - It gives developers more computing power and data storage as they need.
    - Hide the complexity of data and information flow.
    - Better documentation and tighter quality control.
- Cons of DevOps as a service:
    - Requires a specific level of software development expertise.
    - Cloning an enterprise infrastructure is complex.
    - Security is always a concern.
- DevOps tool is an application that helps automate the software development process and helps reduce manual efforts.
    - QuerySurge:
        - Integrates into the DevOps pipeline.
        - Verifies large amount of data quickly.
        - Detect code changes.
        - Provide data analytics.
- Manual infrastructure drawbacks:
    - Configuring infrastructure manually is very error prone.
    - Time consumption
    - Dependency.

- Terraform is that it does not ask you how to get from the infrastructure you have to the infrastructure you want.
    - Configure multiple components.
    - Terraform community.
    - Destroy environment.
    - Add terraform to an existing infrastructure.
- Chef and Puppet are configuration management tools. They are designed to configure and manage the software that is running on a machine (infrastructure) that already exists.
- Containers:
    - Without containers, the dependencies and files are all placed together on a server. Since managing these dependencies is time consuming, similar apps are typically grouped on the same server, sharing their dependencies. Containers solve this problem since each app will run inside its own container with its own dependencies.
- Container schedulers:
    - It makes sure that resources are used efficiently and within constraints.
    - It makes sure that services are (almost) always running.
    - It provides fault tolerance and high availability.
    - It makes sure that the specified number of replicas are deployed.
- Kubernetes make sure that each container is where it’s supposed to be, and that the containers can work together.

Cloud-native lecture:

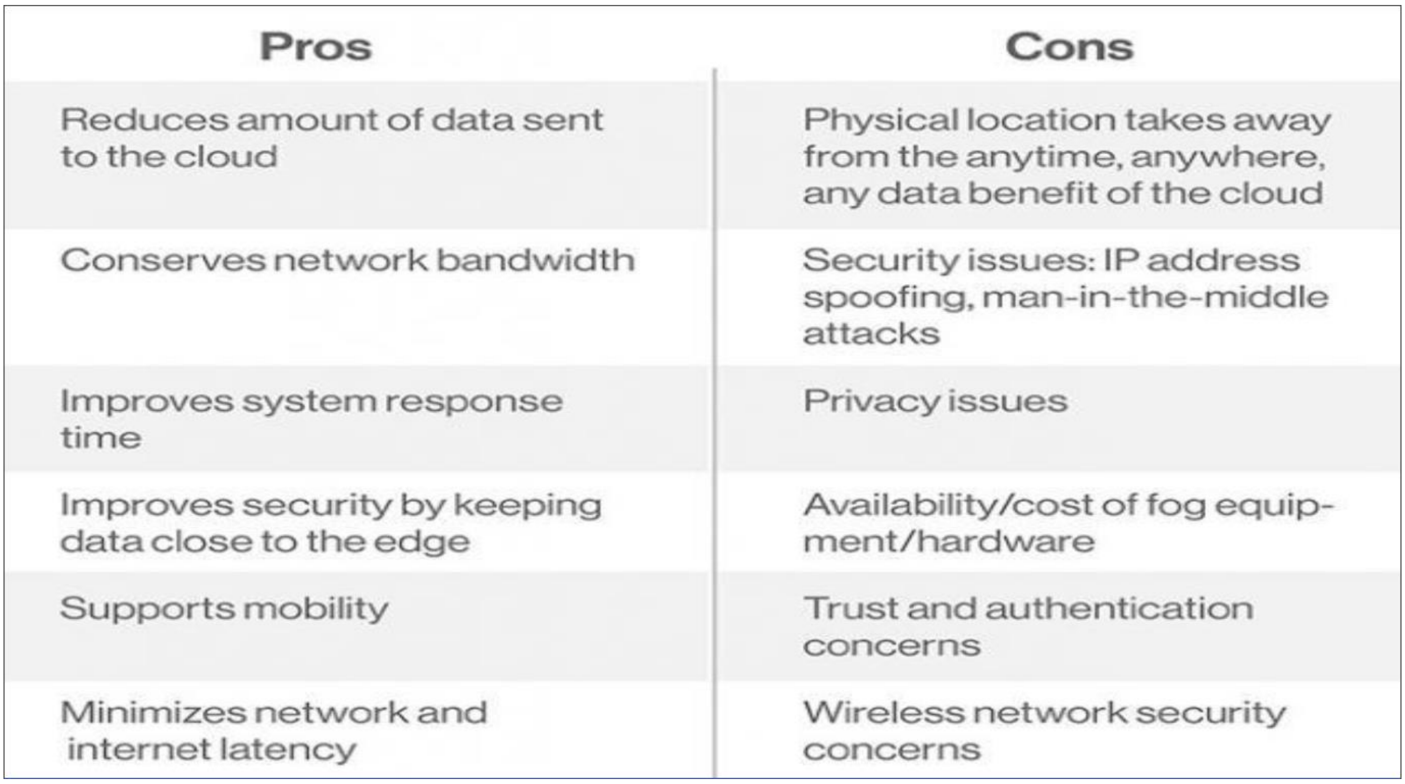
- It is modern approach to build and run software application that exploits the **flexibility** and **scalability** of cloud computing. and run application in dynamic environments such as public, private, and hybrid clouds. They are often built from smaller independent pieces, called **microservices**.
- Reasons for using cloud native:
    - Speed.
    - Agility.
    - Scalability.
- Cloud native a little more narrowly, focusing on application containerization
    - Applications are broken down into **microservices**.
    - Packaged in lightweight containers to be deployed and be used across a variety of servers.
- Microservices: is an architectural approach in which a single application is composed of many smaller, loosely coupled and independently deployable components or services. These services communicate with each other via a combination of **API**.

- API (Application Programming Interface) which is a software intermediary that allows two applications to talk to each other (linking between multiple applications or application and services).
- A cloud application programming interface (cloud API) enables applications to communicate and transfer information from one to another in the **cloud**.
- There are four key layers you'll need to consider when planning your cloud
- architecture:
  - Information management layer - your data repositories.
  - Application layer - where your applications live.
  - Integration layer - where APIs connect your services.
  - Interaction layer - where your API gateway enables interaction between services and a client.
- **API gateway** sits between backend services and a client (requester) to transmit requests and responses.
- **API orchestration** typically requires creating a single API that offers valuable functions to its consumers, often by making multiple calls to multiple different services to respond to a single API request.
- **API catalog** allows you to strategically manage, promote, and share APIs with relevant developers and end users.
- **API integration** to integrate your applications together. So, we will just need API platform to do this like (**AKANA**). It ensures you can connect your cloud applications easily, create new cloud APIs, and work with your existing data sources.
- Cloud APIs are often categorized by type:
  - **PaaS APIs**: Platform as a Service APIs provide access to backend services such as databases.
  - **SaaS APIs**: Software as a Service APIs facilitate connections between cloud services at the application layer [Google maps API].
  - **IaaS APIs**: Infrastructure as a Service APIs enable cloud-based compute and storage resources to be provisioned and de-provisioned as quickly as possible.

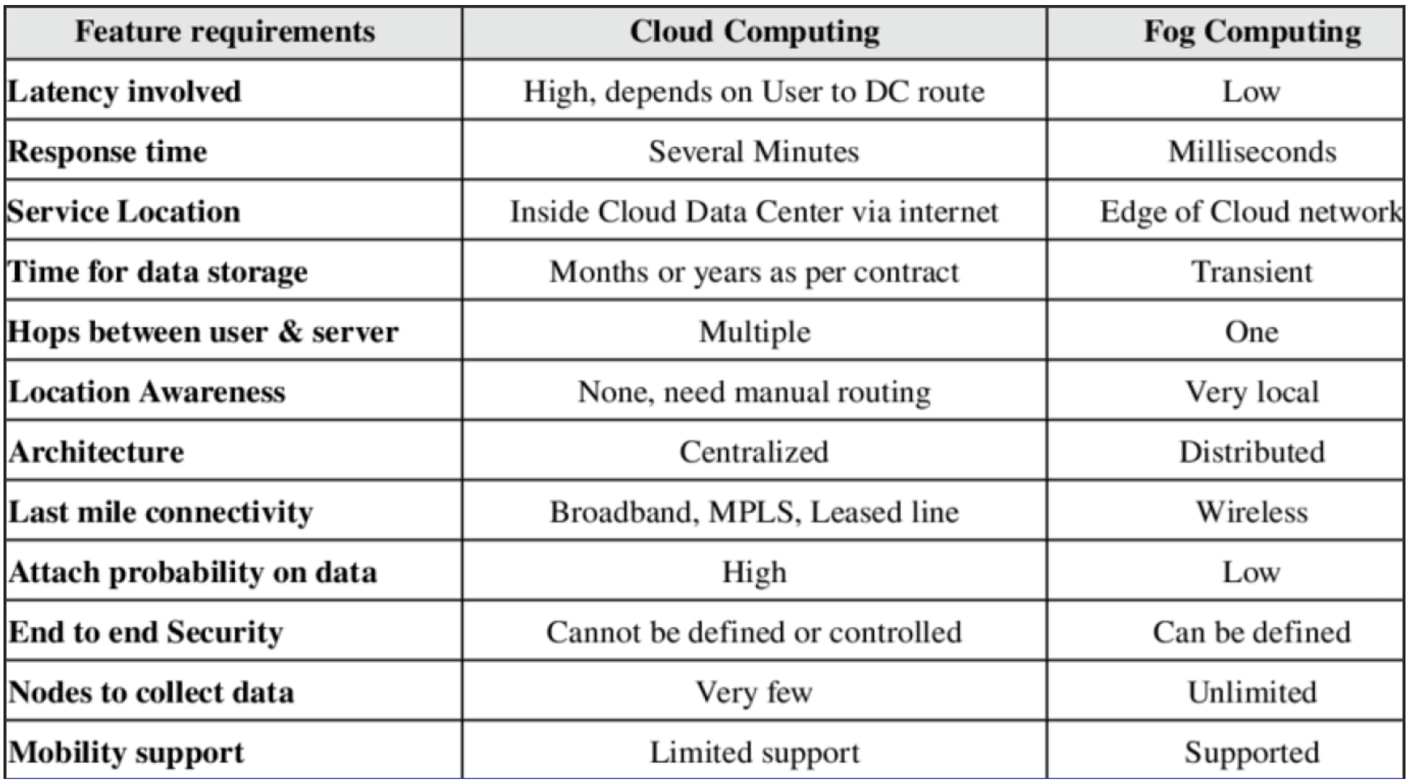
Edge/frog computing lecture:

- Edge computing is a distributed computing framework that brings enterprise applications closer to data sources such as IoT devices or local edge servers.
- Edge computing is an optimization of the cloud, instead of relying on the cloud to do all the work edge computing does the compute work on site sometimes on the edge device itself.
- An edge device is a device that controls data flow at the boundary between **two** networks.
- A common misconception is that edge and IoT devices are synonymous.

- An IoT device is a physical object that has been connected to the internet and is the source of the data. An edge device is where the data is collected and processed.
- Evolution of computing:
    - Early computing.
    - Personal computing.
    - Cloud computing.
    - Edge computing.
- Advantages of edge computing:
    - Reduce latency.
    - Speeds processing.
    - Reliability.
    - Autonomy.
    - Scalability.
- Edge computing examples:
    - Manufacturing.
    - Farming.
    - Network optimization.
    - Improved healthcare.
    - Transportation.
    - Retail.
- Challenges of edge computing:
    - Limited capability.
    - Connectivity.
    - Security.
- Fog Computing is a decentralized computing infrastructure in which data, compute, storage, and applications are located somewhere between the data source and the cloud.
    - Fogging enables short term analytics outside the cloud datacenters.
    - Fog computing contains various devices like fog computing gateway.
    - Most sensitive data are handled by nearest fog node to end device.
    - Less sensitive data are sent to aggregate node for analysis and sends the resulted decisions to the nearest node.
- Applications of Fog computing:
    - Linked vehicles.
    - Smart cities.



Pros	Cons
Reduces amount of data sent to the cloud	Physical location takes away from the anytime, anywhere, any data benefit of the cloud
Conserves network bandwidth	Security issues: IP address spoofing, man-in-the-middle attacks
Improves system response time	Privacy issues
Improves security by keeping data close to the edge	Availability/cost of fog equip-ment/hardware
Supports mobility	Trust and authentication concerns
Minimizes network and internet latency	Wireless network security concerns



Feature requirements	Cloud Computing	Fog Computing
Latency involved	High, depends on User to DC route	Low
Response time	Several Minutes	Milliseconds
Service Location	Inside Cloud Data Center via internet	Edge of Cloud network
Time for data storage	Months or years as per contract	Transient
Hops between user & server	Multiple	One
Location Awareness	None, need manual routing	Very local
Architecture	Centralized	Distributed
Last mile connectivity	Broadband, MPLS, Leased line	Wireless
Attach probability on data	High	Low
End to end Security	Cannot be defined or controlled	Can be defined
Nodes to collect data	Very few	Unlimited
Mobility support	Limited support	Supported