



uting

and computing resources (networks, servers, storage, apps, services) everything from applications to data centers over the internet on a pay-for use basis.

- **A shared pool of configurable computing resources (NIST)** : networks , storage, servers , apps and services
- **Hypervisors role** : an open source VM monitor enable multiple OS to run alongside each other sharing the same physical computing resources and separate VMs logically assigning each a share of computing power ,memory and storage
- **Cloud benefits that lower risk for Enterprises** : pay as you go , the speed with which apps can be up and running on the cloud

► Characteristics :

1. **On demand self-service** : get access to cloud resources using a simple interface without interacting with the service provider
2. **Broad network access** : cloud computing resources can be accessed remotely via laptops, phones, tablets , workstations ...
3. **Resource pooling** : computing resources can serve multiple users and dynamically assigned to customers
4. **Rapid elasticity** : resources are elastically provisioned
5. **Measured service** : you only pay for what you used (pay as you go)

► Deployment models :

- **Public** : many users on the same cloud hardware who don't own the resources (water, electricity, gas) and offers economies of scale and highly reliable. Risking security, used for backup.
- **Private** : single usage for the organization, owned and managed by the owner of the infrastructure (org or cloud provider). VPC is a pub cloud offering that allows the organization to establish its own private cloud, it provides the benefits of the pub cloud without open and shared platform disadvantages, commonly used to integrate data from existing apps, flexibility to build and move apps anywhere, more secure.
- **Hybrid** : gives the organization the freedom and flexibility to choose which cloud deployment suits which of their features and apps. Interoperable , scalable and portable between cloud service providers , used for SaaS.
- **Hybrid monocloud** : one cloud provider offers both pub and priv services.
- **Hybrid multi-cloud** : uses multiple pub cloud services from different providers to leverage the best features of each service.
- **Hybrid composite multi-cloud** : multiple pub clouds and optional private cloud.

► Service models :

- **IaaS (leasing a car)** : get access to infrastructure and physical compute , network and storage resources that the user has significant control on , pay as you go , (IT admins , developers)
- **Concerns** : lack of transparency and dependency on a 3rd party for workload availability and performance .
- **PAAS (renting a car)** : less interaction with the infrastructure, shared maintenance responsibility , pay as you go , (dev , devops teams)
- **Concerns** : information security threats by depending on the service provider's infrastructure, customers lack control over the provider's changes
- **SaaS (taxi)** : low user control , maintenance responsibility of the provider , subscription , (for end-users and business teams)
- **Concerns** : data ownership and safety , 3rd party maintains business-critical data, needs good internet connection .

► Modes of storage :

- **File ,Block and Object** : object is the most common mode, it is highly distributed and resilient.

- **AZ** : availability zones , each zone has its own power , cooling and networking resources
- **Cloud data center** : a huge warehouse containing cloud infrastructure such as servers , networking equipments and other physical tools
- **Storage** : bare metal servers and VMs are provisioned with default storage on the local drive however they can be deleted by customers and thus the loss of data but the customer can preckle the importance of this data , how fast and often he accesses it so that data can be persisted using other data storage options (storage modes)
- **Networking infrastructure** : includes routers, switches ... some are virtualized using API leading to easier provisioning configuration and management.
- **Networking servers** : need two interfaces to be configured . *Public interfaces* that connect the servers to the public internet. *Private interfaces* provide connectivity to our other cloud resources and keep them secure.
- **Networking precautions** : setting security groups and ACLs (access control lists), VLANs, VPCs, VPNs... to configure which network traffic and users can have access to our resources. Also hardware appliances like FW, load balancers, GWs , traffic analyzers ...
- **CDNs** : content delivering networks given by cloud providers makes users access to the content faster world wide by getting it closer to them

► Virtualization

- **Benefits** : cost savings due to multiple virtual env on one piece of infrastructure, agility speed due to easy provisioning, lowers downtime by easily moving vms from one HV to another when a host is down.
- **Hypervisors** : a piece of software that runs above the physical server, they pull the resources from the phy server and allocate them into our virtual environment
- **Type 1 hypervisor (bare metal hypervisors)** : directly installed above the phy server most secure and used
- **Type 2 hypervisor (hosted)** : there's a layer of host OS between the HV and the host such as oracle virtual box...
- **Types of VMs** :
- 6. **Shared/public cloud VMs** : pre-defined storage, hourly seconds and monthly billing
- 7. **Transient or spot VMs** : take advantage of unused capacity in a cloud data center which lower costs but can be reclaimed by the cloud provider at any time thus losing these VMs but good for testing and dev apps and running high performance computing workloads at low costs
- 8. **Reserved virtual server instances** : reserve capacity and guarantee resources for future deployments and choosing the duration
- 9. **Dedicated hosts** : single tenant isolation, only my VMs can run on a given host so they can exclusive use of resources and capacity, specify the data center and the pod , allows for max control over workload and used for meeting compliance and regulatory requirements
- **Bare metal server** : is a single tenant phy server, the CP manages the server up to the OS and if it goes down the CP will fix it and reboot the server, preconfig by CP or custom-config as per customer specifications (processors/RAM/hard drives/OS ...), the client can install his own OS and new HV, takes longer time to provision since they are

nsive, only offered by some CP, dedicated to long term use, good alternative in the cloud for apps that require high security control



- **Containers** : exe unit of software in which apps code is packaged so that the apps can be run anywhere (desktop/cloud/ traditional IT), small fast and portable and do not need to include a guest OS on every instance, they use less resources, they are lightweight and consume fewer resources than Virtual Machines

► Cloud storage

- The faster the read/write speed the higher the per gigabyte cost
- **Direct attached** : local storage , fast, stores usually server's OS. Not great for uses other than OS storage bcz it's temporary (lasts as long as the compute resource is attached to), can't be shared with other nodes and not resilient to failure
- **File storage** : NFS (network file storage), the storages are connected to compute nodes via ethernet network, slower than direct attached and block storage bcz the data travels via an ethernet network, more secure data, more resilient to failure, lower cost than both of them, can be used on mult servers or nodes at once which is good for common workloads , used for workloads where consistent high speed is not required, the higher the IOPS (I/O operations per second) which is the speed at which the disks can read/write data , value the faster the speed of underlying disk the higher the cost.
- **Block storage** : it breaks files into blocks of data and stores each block separately under a unique address, secure data, read/write speed faster than file storage, higher price, perfect for workloads that need low-latency such as databases, not suitable for shared storage between mult servers, IOPS same as FS, mounted to a compute node via high speed fibre network, persistence (the storage doesn't vanish when the compute node it is related to is terminated) however we can set the storage to be auto deleted with the compute node it was linked to. Taking snapshots that are a point in time image of the storage (backup ES and FS), they are fast to create, don't require downtime and record only changes to data, cannot be used to recover individual files.
- **Object storage** : accessed via an API instead of compute nodes, the cheapest, slowest, infinite in size, uses buckets to store data they're like a folder that has metadata providing the location of data and the last access, cannot place buckets within buckets, don't need to provide any sizing info cz it will just hold the data and the service provider will ensure that there's sufficient storage capacity available, add data slowly or quickly and shrink down as needed, buckets are assigned depending on the resilience level (data stored in one geographical location when high availability is not required / data stored mult times in different data centers costs more with high availability). Use cases are text/audio/video files IoT data VM img backup files data archives and any static data where fast read/write speed is not necessary, not suitable for running OS, databases or anything that changes. Effective solution for backup disaster recovery.
- **Object storage data tiers** : *standard tier* bucket is where we store objects that are frequently accessed has highest per gigabyte cost, *vault/archive tier* is where once or twice per month accessed data is stored with lower storage cost, *cold vault tier* stores data that is accessed once or twice a year costs a fraction of US cent per GB per month and its retrieval takes hours and costs a lot, *automatic archiving rules* when an object isn't accessed for an amount of time it will auto be moved to a cheaper storage tier, object storage does not come with IOPS options, not suitable for apps with fast access to files.
- **Object storage data API** : S3 compatible API which means dev can create code that will work against multiple vendor's object storage solutions, allows apps to manage object storage and buckets as well as GET and PUT objects to and from them
- **CDN** : distributed server network that delivers copies of website content to users based on their geo location, it reduces the distance between the visitors and the website server which makes the website faster and reduces the load coming to the server and increased security cz data is retrieved from the CDN.

► Emergent cloud trends

- **Hybrid multicloud** : is when different services are coming from different CP. Use cases are cloud scaling on a global level by using hybrid architecture, composite cloud, data and AI, modernization, prevent lock in to a specific to the vendor's platform and having flexibility to move workloads from one cloud platform to another as the need rises.
- **Micro services** : single app composed of multiple coupled and smaller components or services they communicate with each another with API events streaming and msg brokers. App components can be updated and developed by multiple developers working independently, independent scaling.
- **Use cases** : dev breakdown large apps into small parts independently working on each of them, containers are the delivery method for microservices or codes wherever they need to go and if a container is not working it will be replaced with another one without disrupting how the rest of the app functions, these containers find each other by *service discovery* which create a roadmap for micro to communicate using API and the code doesn't have to be written from scratch. The microservice approach helps dev quickly develop apps in parallel and businesses keep up and grow with their customers.
- **Serverless computing** : IBM *cloud functions*, AWS *Lambda*, Microsoft *Azure functions*. It offloads responsibility for scaling, scheduling patching and provisioning management tasks to CP allowing dev to focus on dev and testing, requires no provisioning of servers and runs code on demand, paying for used resources not idle capacity
- **Use cases** : Data and event processing, IoT, microservices and mobile backends. Short-running stateless functions, seasonal workloads, production volumetric data, working with text/audio/img vids and event-based processing are best fits for serverless.
- **Cloud native apps** : app developed to work only in the cloud environment
- **Use cases** : everything that lives in the cloud should have a cloud native app
- **DevOps** : collaborative approach where business owners, developers, operations and quality assurance stakeholders collaborate to deliver a software.
- **DevOps solutions to cloud's complexities** : automated provisioning and installation, continuous integration and deployment pipelines, define how ppl work together and collaborate, test in low cost production like environments, faster recovery from disasters.
- **Application Modernization** helps organizations accelerate the digital transformation, take advantage of new technologies and services, and become more responsive to changing market dynamics. Cloud computing is one of the key enablers of application modernization.

In motion, key management, scan container images before deployment, users access control, network protection. e and threats, turn these insights into actionable intelligence and remediate :ure build and manage security

Engineers : they assist the whole software dev lifecycle (writing-testing-maintaining) in addition to app front and backend. They need to have technical skills (data structures-distributed systems-OS-algorithms-experience with databases- web app dev languages), business knowledge and experience with at least one of the major CP

- **Cloud integration specialists** : integrate new cloud services apps and infra with existing systems and services .
- **Cloud data engineers** : integrate new data management technologies and software engineering tools into existing infra, collaborating with data scientists to create predictive models and proofs of concepts (POCs)
- **Cloud security engineers** : protect organization's systems and app data (CIA), determine security requirements, plan implement and test security systems, recommend innovative technologies to enhance security of the cloud based environments, in-depth knowledge of cloud platforms and services and DevOps tools.
- **Cloud DevOps engineers** : collaborate with dev and ops teams to create reliable and rapid release updates for software, creating custom automation tools, track design bugs and automating this process for dev, monitor security systems, containerization expertise.
- **Cloud solutions architects** : translate business requirements into app architecture and design, in-depth knowledge of cloud platforms and services, software design patterns, DevOps tools, networking, security concepts.