ICT - Infrastructures

1. What is the difference in one sentence between vm and container? (3-TIMES)

Docker provides a differential filesystem, which is a filesystem that is a diff of the host's filesystem, an abstraction where the new "inner" file system is based on a given one, where the system will only store the differences between the original file system and the new one.

One of the key differences between a VM and a container is that containers do not have a **virtual switch**. A container uses the same MAC address of the host. A **VM is safer than a normal process** because it is **isolated**.

2. Which is the most critical part of live migration (vnic forwarding or memory paging)

The most critical part of live migration is: ***memory paging***, **not** vNIC forwarding.

During live migration:

1. **vNIC forwarding** is relatively straightforward:
    - The VM's virtual network interface is rerouted (or proxied) to the destination host.
    - This is done quickly and usually with little impact, thanks to MAC address spoofing and ARP updates.
2. **But memory paging is complex and critical**:
    - The VM's **entire RAM contents** must be **copied** from the source to the destination.

- **Pages may change (dirty pages)** during the copy.
- These changed pages need to be tracked and **resynchronized**.
- If the memory isn't consistent at the destination, the VM will **crash or misbehave** after cutover.

A **NIC** (**Network Interface Card**) is a **hardware component** that allows a **computer or server to connect to a network**— like the internet or a local area network (LAN).

A **vNIC** (Virtual Network Interface Card) is a **software-defined network adapter** that a virtual machine (VM) uses to connect to a network — just like a physical network card (NIC) in a real computer.

3. How many servers for 1 TB/s data ingestion filter and storage (2 TIMES)

1 TeraByte/second = 1,000,000 MegaBytes/second.

100 Gbit/s NIC (network interface card) convert it to MB/s

100,000 Mbit/s ÷ 8 = 12,500 MB/s. per server processing

1,000,000 MegaBytes/second ÷ 12,500 MB/s. = 80 Servers total

To estimate the number of servers for 1 TB/s ingestion, I convert the data rate into MB/s (1,000,000 MB/s). Then, based on the assumption that a high-end server with 100 Gbit/s NIC can handle about 12,500 MB/s of data processing, I divide total data rate by server capacity, resulting in 80 servers. Considering real-world overhead like filtering, redundancy, and buffering, I add roughly 20% headroom, arriving at around 96 servers minimum. This estimate also depends on workload complexity, infrastructure efficiency, and technology choices.

What is the orchestration layer (2 TIMES)

The **Service Orchestration Layer** is the part of the cloud system that connects and manages different services and actions. When a user (called a **tenant**) asks

for a service—like creating a virtual machine or setting up storage—the orchestration layer automatically starts the right steps to make it happen.

This process is called **workflow automation**. Instead of people doing each step manually, the system runs the needed steps in order, by using a special software called an **orchestrator**. This orchestrator is like a manager that knows what needs to be done and makes sure everything runs smoothly.

Cloud providers use orchestration to:

- Provide services to customers (like UniPi using Microsoft Azure),
- Manage and grow their infrastructure (like adding more storage or servers),
- Handle issues, do backups, track usage, and send bills.

Even though some manual help might still be needed sometimes, providers want to **automate as much as possible** to save time and avoid errors.

The **orchestration layer** is the part of a cloud system that **automatically connects and runs different services and actions** when a user requests something. It uses workflows to make sure all the right steps happen in the correct order — without needing people to do them manually.

Having solar panels would affect the PUE?

No, solar panels do not directly affect PUE. It measures **how efficiently a data center uses power** — especially how much of the power goes **directly to IT equipment** (like servers) vs. other things (like cooling, lighting, etc.).

Should you care about the PUE if you have solar panels?

Yes, you should still care about PUE — even with solar. **PUE is a measure of efficiency**, not just sustainability. You may **spend more** on battery backup systems or cooling tech.

Why would you need new 200kw servers (ai workloads)

What is CAPEX and OPEX ?

Term	Full Name	Meaning
CAPEX	Capital Expenditures	One-time costs to **buy or build** physical assets
OPEX	Operational Expenditures	Ongoing costs to **run and maintain** a service or system

1. What is PUE? (2 TIMES)
2. What is CRAC cooling? How is it different from InRow cooling?
3. What is LACP?
4. What's RAID? RAID 1,5,6
5. What is NvME?
6. What's a LUN.
7. SAN, NAS
8. Why we use virtualization in servers?
9. What's a Hyperconverged infrastructure?
10. WHAT IS HPC ?
11. What is the main problem of SAN architecture in storage? (Bandwith)
12. How would you design a fibre channel SAN with 100Gb/s input

1. Would VM replicas affect SLA?

-
- Cooling system of a physical DC affect the service catalog and SLA defined by the private cloud that runs on the physical infrastructure ?
- What can we say about the service catalog?
- How can service catalog be affected by physical constraints? Example of service that you can run with 15 kwatts per rack of power?
- What's a Full Fat Tree
- What's Spine and Leaf architecture? Is there a limit of leaf in a network? What can i do if i finish the spine's ports?
- What's thin provisioning?
- What happen when you do a snapshot of a VM? How is the drive snapshoted?
- What's Capacity management/planning? How CAPEX and OPEX are related to it?
- We want to implement a storage service with hundred thousand users how do you think about the infrastructure? Tell me about storage, servers and cooling.
- How is possible in an hypervisor to overbook the memory? How can you guarantee that a VM does not go in space memory of another VM.
- Why Monitoring is important?
- What are the strategy to build a HA system?

Data Center

Datacenter - active systems that allow to host server storage and network.

1. **Enterprise Datacenter:** typically housed within dedicated buildings and are custom-built for the organization’s needs.
2. **Hyperscale Datacenter:** a very large data center designed for massive workloads and high scalability**,** often used by companies like Amazon, Google,

and Microsoft, assembled at factory and pre cabled. These facilities are engineered to handle enormous amounts of data and computing resources, allowing for rapid scaling and adaptation to changing demands. Its not possible to follow those hyperscaler datacenters in your own one, since they all have at least 5 backups to keep the servers on in case of failures.

In the structure of data center we have **Racks**. A data center rack is a type of framework that is usually made from steel and houses your servers, cables, and other equipment. Your servers can fit neatly into the framework, helping to keep them organized and safely off the floor. Racks are made of 42 Units.

Besides server theirselves, there is a cooling system. The first issue is the how to provide cool air. Then there is also how to define an evacuation plan

Cables are not super-resistant to current. A lot of current passing through a copper wire will exhaust both the wire and the components receiving such current; hence the current should also be balanced among different cables, to avoid exhausting some components before the others.

Hardware tend to fail, in case of datacenter everyday something breaks.

Smart technology: predictive failure system, tells that there are signs that there will be a failure on a drive. Can proactively substitute the drive, recycling keeps costs under control.

POD (Point of Distribution) is a modular unit in a datacenter that bundles **compute, storage, networking, power, and cooling** into a self contained, scalable block. It enables efficient deployment, fault isolation, and automation in large-scale infrastructure.

Typically in datacenter you have **power room** full of switches that distributes the power. They distribute power in high voltage 380V.

A **UPS** —first of all— stabilizes the output current.

UPS: Uniform power supply stabilize power, extend lifetime of datacenter components. Give power from generator or main line. Without batteries it is impossible to power on the generator and keep everything up and running. Batteries meant to sustain datacenter during switch from grid to generator.

PDUs stands for Power Distribution Units, and allow to distribute power for server units inside racks. Typically, for each PDU there is another one, providing

redundancy and thus resilience/robustness. Multiple racks may be powered by the same PDU, as it happens in SPG.

The UPS is attached to the PDU (Power Distribution Unit) which is linked to the server. As briefly stated before —for redundancy reasons— a server is powered by a pair of lines, that usually are attached to two different PDUs. The server uses both the lines, so that there will be continuity in case of failure of a line. In the server there are the power plugs in a row that can monitored via a web server running on the rack PDU.

The power factor is the ratio between the real power and the apparent power.

Power Usage Effectiveness (PUE) measures the efficiency of a Datacenter.

$$PUE = \frac{Total\ energy}{ICT\ energy}$$

The reason for improving Datacenter design is to lower the PUE; basically to save money, but also for "green environment" concerns.

When should PUE be measured ? Generally its calculated as the average of one year.

In particular is strongly unrecommended to build datacenter in geographical zones with high temperature variations over the year. A counterintuitive example is the desert, where the temperature is very high during the day and very low during the night, but in general the temperature over the year is stable; allowing for defining physical processes exploiting such stability. Also the oceans have a very stable temperature; not on the surface, but deep down it is very stable.

- **Operational costs expenditure (OPEX) =** pay for used resources. Service 1/3 of investment is opex every year. Usually maintenance in hours, in case of failure will service in hours.
- **Capital expenditure (CAPEX) =** initial cost. Datacenter is mostly a Capex, lifetime is at least 10 years, you build and sell the service.

Inside the datacenter **Direct Current (DC)** is preferred, because a DC power architecture contains less components (hence less heat production, hence less

energy loss), but the problem is that the power companies supply our buildings with
Alternating Current (AC). AC is more efficient for power companies to deliver, but when it hits the equipment's transformers, it exhibits a characteristic known as reactance. Reactance reduces the useful power (watts) available from the apparent power (volt-amperes).

A "**floating floor**" in a data center, also known as a "raised floor", is a type of construction used in data centers to create a void between the actual concrete floor and the floor tiles where the servers and other equipment are located. This space is typically used for routing cables and for air circulation, which helps with cooling the equipment."

Data Center: Cooling

Cooling is critical since its the most important factor in determining the PUE. Today the standard is air based, but there are some experiments for water based cooling.

Not all chilling techniques are possible in all places, because the temperature of the water must be lower than the one of the air, and the air must be dry.

Note that racks are always placed back-to-back, because the front requires cool air, and the back outputs hot air.

CRAC Cooling System (Computer Room Air Conditioning)

- In this system, **chillers** (cooling machines) take the **hot air** from the top of the room and push **cold air** into the bottom.
- The cold air is pushed **under the raised floor** (called "floating floor").
- Then it comes up through **grates** placed in front of the server racks.
- Each **rack**:
  - **Takes in cold air** from the front.
  - **Pushes out hot air** from the back.

Problems with this system:

1. **Hot and Cold Air Mix Together**
    - It's hard to keep the cold and hot air separated.
    - When they mix, **cooling becomes less effective**, wasting **energy and money**.
2. **No Cooling Based on Need**
    - If one rack is working harder and needs more cooling, the system **can't focus** cooling only on that rack.
    - Instead, it sends **more cold air to the whole row**, even to racks that don't need it.
    - This is **inefficient**, especially when workloads are different across racks.

Who still uses it?

- Almost **no one uses this system today**.
- **Aruba** still does, because their workloads are **very similar** (homogeneous), so the cooling inefficiency is less of a problem.
- **Data centers now use doors and other techniques** to separate hot and cold air more effectively.

TDP = thermal design power, thermal design of a CPU

InRow Cooling:

- **InRow cooling** places cooling units **between the server racks** instead of at the edge of the room like in CRAC systems.

Why is InRow better?

1. **Targeted Cooling (Supports Mixed Workloads)**
    - It can cool **specific areas** where racks are hotter.
    - This is perfect for **heterogeneous workloads**, where not all servers work equally hard.

2. **Efficient Airflow**
    - The server rack **pushes hot air backward**, and the **InRow unit is placed behind to suck in that hot air**, cool it down, and **blow cool air back to the front** of the racks.
    - This creates a **closed loop**, **saving energy** by avoiding wasted airflow.

Smart Fans = Dynamic Infrastructure

- The **fans in InRow coolers** can **change speed** based on **how hot the servers are**.
- Temperature sensors in front of servers help them adjust in real-time → this is called **dynamic infrastructure**.

Chilling Water Outside

Cooling System Design

- **Pipes under the raised floor** carry cold water from chillers (cooling machines).
    - Putting them **under the floor** is safer than on the ceiling (in case of leaks).
- **Chillers outside the building** cool water and send it inside, where **InRow or CRAC units** use it to cool air.
- **Important:** Make sure the water **doesn't heat up too much** while traveling from outside → wasting cooling energy.

Temperature and Energy Saving

- Example: In SPG datacenter, chillers cool water to **18°C**.
    - Even though it seems high, it works well because the **datacenter is designed to run safely up to 26°C**.
    - **Higher allowed temperatures = more energy saved**, because chillers don't need to overcool.

Free Cooling with Adiabatic Chillers

- If **outside temperature is below 18°C**, SPG can **cool water without using compressors**.
    - This is called **free cooling** and saves a lot of energy.

Humidity Matters Too

- If the air is **too dry**, it can cause **static electricity or condensation**.
    - This can **damage racks and even harm people**.
    - So, **humidity must be carefully controlled** in the datacenter.

Active-Passive means that aside from the active system, there is a mirrored one which is shut down waiting for failure and boots up "just in case". This approach is usually not the ideal one, because the second system is very unlikely to be used and is costful. Besides, there are two critical issues with Active-Passive:

1. There is a non-negligible time interval where the switch from the active broken system and the passive one has to happen.
2. If when booted the backup system reveals itself to be flawed and not working, well... very sad /

Active-active systems are usually better, because they also allow for load balancing. In case of SPG there are three cooling systems, and in case one breaks, the other two can keep working. Active-active costs even more, but it is the standard way to go.

High-end CPUs heat up so much that it has became unreasonable to cool them using air.

However, note that water conducts electricity, so a flaw in a water powered cooling system may lead to consistent damage and possible fires.

Oil instead doesn't conduct electricity, and there are some systems which are submerged in oil, but there are *two drawbacks*:

1. Price: oil is way more expensive than water
2. Servicing: it is impossible to maintain the system's hardware.

Distilled water is not conductive, but even not considering that distilling it is expensive, it is impossible to guarantee that is stays pure when traveling in pipes, chillers, and so on.

Most datacenter tend to have an mixed approach to cooling, called **air-to-liquid**. The idea is simple: It is acceptable to use cool air to chill water, which is then used to chill the air by InRow coolers, which chills the liquid which chills the CPUs.

"Can't we simply chill the liquid and send it directly onto the CPUs ?" No.

- Required pressure is different
- Required temperatures do not match
- Having water directly inside the datacenter is risky

Liquid Cooling: is a method of removing heat from datacenter equipment using liquids instead of (or in addition to) air. Traditional air cooling struggles as servers get denser and hotter. Liquids absorb and transfer heat much more efficiently than air.

Spilling Pipes: **Liquid cooling** systems manufacturers allow customers to ensure that their pipes are not spilling by injecting in the pipes a known gas at a known pressure. The customer can measure the pressure when the product is shipped and check whether it is the expected one, and if not, send back the device.

Handling spills is an open problem. Theorically, the idea would be to check for pressure variations, but this is currently impossible to be done on each entrance of each rack. Too much actuaction and sensoring would be required.

Besides, in case a pipe is spilling, the operators must act quickly, before the water spills onto other racks and cause critical damage.

Now fully Liquid cooling in datacenter is risky option, but in the 2-3 years it can be changed totally.

Data Center: Cabling

Storage and Bandwidth in Datacenters

- In modern datacenters, each server is usually connected with a **25 Gbit/s link in both directions**, giving a **total of 50 Gbit/s** bandwidth.
- However, **SSDs are much faster than that**. A single NVMe SSD can reach about **3.5 GB/s**, so just **4 SSDs can already saturate a 100 Gbit/s link**.

Shift in Bottleneck

- In the past, the **slow speed of hard drives (HDDs)** was the biggest bottleneck.
- Today, the bottleneck is **network speed**, because **SSDs are much faster than what the network can handle**.

HDDs vs SSDs

- **HDDs** are still used for **cold storage** (data that's rarely accessed).
- **SSDs** are used for **fast access**, especially for CPUs and high-performance workloads.
- Some servers now come with **only SSDs (flash storage)** already built-in.

Cost vs Performance

- SSDs are more expensive than HDDs, but in high-end servers, **this cost is not a big deal**.
- These systems already use **top CPUs, GPUs, and RAM**, and you don't want to **waste energy and time** by waiting on slow drives.

SSD Lifetime

- SSDs do have **a limit on how many times you can write to them**, but:
  - If you write the **entire disk once per day**, it will still **last about 5 years**.
  - By that time, most companies would **replace or upgrade hardware anyway**.
  - SSD failures are also **predictable**, so they can be **planned for**.

Cabling one of the most complicated parts of datacenter. It indicates:

- Point to point connection
- Device to manage

Patch panels = connect different parts in a single point. This is **structured cabling**

In HPC cabling is in the front of the server, not in the back. Because you want to be able to change, not affect the rest of the system.

Cabling should be:

- **Ordered:** While cabling - *order cabling using the sides* of the racks so that the air flow is not obstructed.
- **Documented:** Important to document every single cabling, since you will forget everything, even in small technical rooms inside the building. Documenting means all the cabling is documented point by point, so that by browsing this particular open source system that we use to keep track of things, I can virtually follow all the cables to check the physical connectivity without having to go inside the data center.

Cables and Standards

A **bottleneck** is the **weakest or slowest part** of a system that limits overall performance or throughput

Electric current (or signals in copper cables) **travels at around 0.6 times the speed of light** (s ≈ 0.6c), depending on the cable type. In comparison, signals in ***optical fiber*** travel at about **0.67c to 0.7c** — so in theory, **optical fiber is slightly faster** than copper in terms of signal propagation speed.

A **NIC** is a hardware component (usually a small circuit board or chip) that allows a computer, server, or device to **connect to a network** — either via **Ethernet (copper)** or **optical fiber**.

Note that **optical fiber connections usually require two fibers**:

- One for **transmitting (TX)**

- One for **receiving (RX)**

The most common connector types are SC (larger, click-in type) and LC (smaller, more compact).

In many cases, the connectors at the ends of the cable are **detachable**, allowing you to **swap the fibers**. This is useful when the **TX and RX need to be flipped**, for example, if the signal is not working due to a mismatch.

When transmitting data using electricity, several factors must be considered:

- **Interference** from nearby cables or devices can distort the signal.
- **Cable size and length** affect how well the signal travels — longer or thinner cables can weaken it.
- Also, switching from a **1 (high voltage)** to a **0 (low voltage)** isn't instant. It takes time, and that delay can impact transmission speed and accuracy.

RJ45 is a standard physical connector used for **copper Ethernet cables**, commonly supporting speeds up to **1 Gbit/s**.

Even Cat 7 **cables**, which can support up to **10 Gbit/s**, still use the RJ45 connector. However, Cat 7 cables are **very thick and stiff**, which makes them hard to bend and install in tight spaces.

It's estimated that around **70 billion meters (70×10^9 m)** of Ethernet cable have been installed globally — making it the **most widely used cabling type** in the world.

A transceiver is a device that can both:

- **Transmit** data
- **Receive** data

That's why it's called **trans** + **ceiver** (transmit + receiver).

Sometimes, fiber cables have **different connectors on each end**, such as **LC on one side and SC on the other**, to match the ports of different devices.

Instead of having built-in fiber ports, devices use **SFP ports** where you can **plug in a transceiver**.

These transceivers, such as **SFP modules**, are small devices that:

- **Receive electrical power and signals** from the switch or server
- **Convert** those electrical signals into **optical signals** (for sending through fiber) — and vice versa

The main purpose of the **SFP standard** is to **decouple the optical part (transceiver) from the hardware**, such as servers or switches. This makes systems more **flexible**, **modular**, and **easier to upgrade or repair**.

They allow to go optic-copper, copper-optic, optic-optic
and copper-copper.

- **SFP Port**
  - This is a **slot** on a switch, router, or server
  - It's like a **USB port**, but for network transceivers
  - The port itself does nothing until you plug something into it
- **SFP Transceiver (Module)**
  - This is a **small pluggable device** that goes into the SFP port
  - It **converts electrical signals** from the device into **optical signals** for fiber — or keeps it electrical for copper
  - It has connectors (like LC or RJ45) to plug in network cables

SFP −−> 1Gbit

SFP+ −−> 10Gbit

SFP28 −−> 25Gbit
This is the current standard

QSFP28 −−>4×25Gbit

Data traffic is always at least SFP+. Current standard is SFP28. Various SFP are typically compatible, the shape of the plug should stay the same. On switches there also some ports which are QSFP+ or QSFP28, which allow up to 40 and 100Gbit/s respectively, and are used for north-south traffic.
The Q letter stands for Quad

Switches for datacenters should be non-blocking, meaning that no port has to wait for other ones —or any other thing— before transmitting, they can also transmit simultaneously.

What is InfiniBand?

Ethernet is the most common way computers connect in networks, but InfiniBand is another system used mostly in supercomputers and very powerful computers. InfiniBand can send data much faster and with less delay — it takes about 2 millionths of a second (2 microseconds) for data to travel. It can also send very large messages (up to 2 gigabytes) and can prioritize different types of data with 16 priority levels. InfiniBand makes sure that if data arrives, it is complete and not broken. Unlike Ethernet, it does not use the usual internet communication system (TCP/IP) but uses a simpler, faster method called MPI, which helps computers talk to each other directly.

How is InfiniBand used?

InfiniBand cables and plugs look like Ethernet cables but they don't work with Ethernet devices. Computers need special InfiniBand cards and switches to use it. Usually, only the most important servers in a system connect with InfiniBand. These connect to a special InfiniBand switch, which then connects to the rest of the network using normal Ethernet because Ethernet is cheaper and works well enough for less important data.

What is Omni-Path?

Omni-Path is a fast computer network technology made by Intel. It is an improved version of Intel's earlier technology called InfiniBand. Omni-Path competes with the fastest InfiniBand versions, known as EDR and HDR.

Why is it Important?

Intel is building Omni-Path to help create supercomputers that can reach “exascale” speeds. Exascale computers are very powerful machines that can do a billion billion calculations every second — the next big goal in supercomputing.

What is RDMA?

RDMA stands for Remote Direct Memory Access. It’s a special technology (not a network protocol) that lets one computer read or write data directly into another computer’s memory without using that other computer’s CPU or operating system.

Why is RDMA Useful?

RDMA helps move data very fast because it skips extra copying steps. Normally, data has to be copied from the application to the operating system and then sent over the network. RDMA allows the network card to send data straight from the application’s memory, saving time and reducing delays. This is very useful for things like distributed storage, where data needs to move quickly between computers.

What is RoCE?

RoCE means RDMA over Converged Ethernet. It is a protocol that lets RDMA work over regular Ethernet networks, so computers can use this fast memory access method even on common network setups.

Important Note

While RDMA makes data transfer very fast, it can create security risks because it lets one machine access another machine’s memory directly without going through normal checks.

Networking

The two key aspects of a network are:

1. **Bandwidth** → amount of data per second that can be moved through a specific connection

2. **Latency** → is the amount of time required for trasmitting data, measured from the moment it is sent from the source to the one it is available to the source.

In a datacenter, sending data over an Ethernet cable takes about 0.5 microseconds (very fast). But if you use the usual internet communication system called TCP/IP, the delay (latency) goes up to about 70 to 90 microseconds.

Modern hard drives are so fast that this delay can slow down how quickly the CPU and drives work together.

Sometimes, multiple cables are combined (for example, four 10 Gbit/s cables combined to make 40 Gbit/s). This works only if the cables are joined at a very basic physical level. If not, the TCP/IP system will only use one cable at a time, so you don't get the full combined speed.

Some computer tasks are very **sensitive to delays**, so the extra time caused by the TCP/IP system can cause problems. Inside modern datacenters, the usual delay is less than one microsecond, which is very fast.

To handle this, special technologies help reduce delays. InfiniBand is one such technology used in supercomputers that provides very low delay. Omni-Path is a similar technology made by Intel that can handle larger systems and is the next version after InfiniBand.

Other technologies like RDMA and RoCE let computers access each other's memory directly without using the CPU or operating system, which skips the slow TCP/IP system.

Fibre Channel switches connect storage devices to server CPUs inside datacenters but are not used for general networking between computers.

SDN stands for Software Defined Networking. It's a new way to control computer networks using software instead of relying only on hardware devices. With SDN, special programs called controllers can talk to the network equipment and decide how data should move around.

Why Use SDN?

Normally, networks are made of many hardware parts that are hard to change or program — they are “ossified,” meaning stuck or rigid. SDN lets us control and program the network easily without breaking or changing the existing setup. This makes networks more flexible and able to adapt to what applications need.

Software Defined Datacenter

When this software-based control is applied not just to the network but to the whole datacenter (including computers and storage), it is called a Software Defined Datacenter. This lets companies offer IT resources like a service, making everything easier to manage and use.

How SDN Works (OpenFlow)

A key idea in SDN is to separate two parts of the network devices:

- The **control plane**, which decides where data should go
- The **data plane**, which actually moves the data

The controller talks to the data plane using a standard called OpenFlow. It can program rules into the Flow Table, which tells the device how to handle different data streams.

Example Use Case: Sandwich Firewall

A university created a clever use of OpenFlow called the Sandwich firewall. When data starts flowing, it first goes through a firewall to check if it is safe. If it’s clean, the data goes straight to its destination. If it’s harmful, the data is stopped and dropped right away.

Hyperconverged Infrastructure (HCI)

HCI is a way to manage computer systems using software instead of traditional hardware. It combines computing, storage, and networking into one system by using something called a hypervisor, which lets you create virtual computers and virtual storage.

How Does HCI Work?

When you add new servers (called nodes) to an HCI system, they automatically join the existing group (called a cluster). The software controls how the computing

power, storage space, and network connections are shared and managed across all the nodes.

Advantages of HCI

With HCI, you only need to deal with one company for support because they handle everything — hardware and software together. This makes fixing problems and getting help easier.

Things to Keep in Mind

HCI nodes can work alongside regular servers that aren't part of the cluster. However, HCI systems produce a lot of network traffic, especially when copying storage data or moving virtual machines between nodes. This can affect network performance.

Programmers usually focus on higher network layers like layer 3 or 4 (these deal with routing and transport). But in datacenters, it's very important to understand how **Layer 2** works.

Why Layer 2 Matters in Datacenters

1. Inside datacenters, most data moves "east-west" (from one server to another) using Ethernet, which works at layer 2.
2. Many important protocols run at layer 2 inside switches, so knowing how layer 2 works helps you understand how the network really functions.
3. MTU (Maximum Transmission Unit) — the biggest size a network packet can have — is a layer 2 setting that affects performance and data flow.

No Routers Doing the Work

Unlike regular networks where routers manage traffic, datacenters often build their own "fabric" network. This means they have to handle layer 2 networking themselves.

Protocols Inside Switches

- **LLDP (Link Layer Discovery Protocol):**

This protocol helps devices in a network discover and understand how other devices are connected. It allows you to partially map out the network's setup and how it works.

- **DCBX (Data Center Bridging Exchange):**

  This is a protocol that helps two devices agree on how to share network resources, especially for storage and data traffic. For example, one device might say, "I need 50% of the network bandwidth to work properly."

  DCBX helps manage this sharing, which is part of Quality of Service (QoS) for Ethernet networks, making sure important data gets the bandwidth it needs.

Common Network Graph in Datacenters

Nowadays, datacenter networks are often arranged like a graph: the inside points (nodes) are switches or routers, and the endpoints (leaves) are servers.

Physical Medium and Data Link Layer

The cables and connections are no longer shared by all devices at once. But, at the data link layer (layer 2), it still acts *like* the devices share the same connection.

On a switch, to imitate a shared connection, it copies the same data frame to multiple ports. But these frames don't have unique IDs at layer 2, so the switch can't tell if a frame is a duplicate.

This can cause a problem called a **loop**, where copies keep circulating and cause a "packet storm," slowing down or crashing the network.

Network Topology: Tree vs. Graph

- To avoid loops, networks are often designed like a **tree** — a structure with no loops.
- But a **fully connected graph** (where every node connects to many others) offers multiple paths for data to travel. This can improve speed and reduce the number of "hops" (steps) before data reaches its destination.

Trade-offs

- Fully connected graphs are expensive and complicated to manage.

- Less connected graphs can cause **over-subscription**, meaning some links don't have enough bandwidth to carry all the data, leading to slowdowns or bottlenecks.

How to Keep a Network Connected but Avoid Loops?

You want your network to be fully connected (like a graph) so data can travel many paths, but you don't want loops that cause problems.

The Solution: RSTP Protocol

RSTP (Rapid Spanning Tree Protocol) helps with this. It:

- Sends signals (probes) to check for loops and find where devices (like PCs) are connected.
- If it finds a loop, it blocks the connection causing the loop, creating a logical "tree" without loops.

Where RSTP Works Well

RSTP is good for networks like campuses or offices where most data moves up and down (North-South traffic).

Why RSTP Isn't Great for Datacenters

In datacenters, most traffic moves side to side (East-West traffic) between servers, and there are many switches and connections. RSTP is too slow (takes seconds to react) to handle this complex setup, so it's not suitable.

Conclusion:

Using RSTP to block loops in a fully connected datacenter network is not a good idea because it can't keep up with the fast, complex traffic there.

Network Chassis Architecture

In the past, a common way to build networks was using a **network chassis**. This is a big box that holds multiple **line cards**—each line card works like a small switch.

All cables from different racks of servers connect to this chassis, and the chassis directs the data traffic to the right place. This setup looks like a **star topology** (everything connects back to one central point).

Advantages of Network Chassis:

- You can start with just a few line cards and add more later, which helps reduce initial costs (CAPEX).
- The chassis acts like a single big switch, making management easier than handling many separate switches.
- It includes **redundancy**—meaning if one line card fails, it can be replaced without shutting down the whole system (hot-replace).
- Redundancy is built-in for every important part, like power supplies and routing modules, improving reliability.

Downside:

Network chassis are expensive because they are complex and modular.

What Came Next:

To save costs, people began stacking regular switches together and using special protocols to make them behave like a single chassis.

Why Chassis Networks Disappeared

When network speeds moved from 10 Gbit/s to 25 Gbit/s and higher, chassis networks stopped being practical.

- It became very hard to design a chassis that could still allow **hot-replacing** parts (swapping components without shutting down) *and* handle these super-fast connections.
- The size of the cables and connectors limited how fast data could move and how well the physical links could work at high speeds.

Because of this, **star-shaped chassis networks became rare** starting around 2013-2014. Instead, switches started to come in fixed, non-modular shapes (you can't add or remove parts easily).

How Switches Work Now

To manage multiple switches together, vendors created special built-in protocols that let two separate switches behave like one. For example:

- **MLAG (Multi-Chassis Link Aggregation)** lets two switches work together.
- They also use **port channels**, which combine two physical ports (one on each switch) into a single logical link.

A port channel is a way to combine two or more physical network ports into one single logical link. This helps increase reliability and overall network capacity.

About Port Channels and LACP

- The official protocol to manage this is called **LACP (Link Aggregation Control Protocol)**.
- A port channel acts like one link, but the bandwidth for a single data stream is *not* combined across the ports.
- Usually, many data streams run at the same time, so the total bandwidth looks bigger.
- Port channels also give **redundancy** (backup connections) and help **avoid loops** in the network.

Three-Tier Architecture

This is an old and simple network design that used to be common but is now mostly outdated. It has three layers of switches arranged like a tree:

1. **Core switches** at the top (the main backbone)
2. **Aggregation switches** in the middle (connect groups of access switches)
3. **Access switches** at the bottom (connect servers and devices)

How It Works

- All the switches connect in a tree shape.
- The **Spanning Tree Protocol (STP)** is used to stop loops, but it also means only one path is active at a time, so backup paths are unused (passive).
- This makes redundancy **active-passive**, not very efficient for traffic going side-to-side (east-west) inside the data center.

- Most traffic must go up to the core switches even if the servers are nearby, causing competition for bandwidth and delays.
- Server-to-server communication often has to cross multiple layers, increasing latency and bottlenecks.

Why It's Not Used Anymore

- It doesn't scale well when there are many switches and links, which is common in modern data centers.
- It can't handle the need to move virtual machines freely between servers efficiently (important for virtualization).
- Newer designs replace this with more flexible and faster architectures.

What is spine-leaf

Spine-leaf is a modern network design architecture used in data centers.

It has two types of switches:

- **Spine switches** – they move data across the whole network.
- **Leaf switches** – they connect to servers and devices.

How it connects

Each leaf switch connects to every spine switch.

This makes sure all devices are only 2–4 steps away from each other.

This keeps network delay (latency) low and predictable.

Not always fully connected

Not every rack (group of servers) connects to every other rack.

It depends on the data center layout. But performance is still good.

No STP problems

Older networks used STP (Spanning Tree Protocol) to avoid loops,

but it only lets one path work at a time.

Spine-leaf uses **ECMP** (Equal-Cost Multi-Path) to use **all paths** and balance the traffic.

This avoids traffic jams.

Easy to expand

If more speed or capacity is needed, you can just add more spine switches.

This is called **scaling out**, and it makes growing the network easier.

What is LACP

LACP stands for **Link Aggregation Control Protocol**.

It combines multiple network cables (links) into one **logical link**.

This helps increase speed and provides a backup if one cable fails.

Why it helps

- It prevents **loops** in the network.
- It gives **more bandwidth** by using several links together.
- The links form something called a **port channel** (a virtual single cable made of many cables).

Limit of a single stream

Even if you join two 25 Gbps cables (getting 50 Gbps total), **a single data stream** can still only go as fast as **one link** (25 Gbps). But in real life, many data streams are used, so the total bandwidth is still higher and useful.

Advantages of Spine Leaf

Modular Design

You can mix different types of switches and easily add more when needed.

Predictable Latency

The delay between any two servers is always the same – usually just 2 or 4 steps (called “hops”).

This helps keep the network fast and consistent.

Bandwidth Control

You can decide how much network space to give for traffic going:

- North-South (to/from outside the data center)
- East-West (between servers inside)

Active-Active Redundancy

All network cables (links) are active at the same time.

If one fails, traffic still flows smoothly through the others.

This is managed by LACP.

No Loops

The structure avoids network loops by design — no need to turn off backup links like in old systems.

Flexible Connections

You can choose how many cables connect the leaf switches (servers) to the spine switches (core of the network), depending on how much speed you need.

Easy Maintenance

You can turn off and replace a spine switch without breaking the network.

The other spine switch keeps things running.

You lose half the speed temporarily, but everything stays connected.

Latency vs. Chassis

Spine-leaf may have slightly more delay (more hops) than old chassis (star) networks.

To fix this, you can use a **very large switch** (like 256 ports) at the spine layer to reduce the number of hops and keep latency low.

What Is Oversubscription?

Oversubscription means **sharing a network connection**.

For example, 10 servers might use **1 shared uplink** to send data to the rest of the network. This can save money and reduce unused bandwidth.

Why Use It?

It works well when devices **don't always use their full bandwidth**.

Instead of wasting fast connections on servers that use little data, we let them share.

When It's a Problem

If many devices **try to send a lot of data at once**, the shared link can get full (a bottleneck).

This causes **slow response times** and poor performance — especially for heavy workloads like:

- Cloud computing
- Big Data
- Virtual machines
- High-speed storage

Best Practice

Devices with high data needs should have **a direct (1-to-1) link** to avoid slowdowns.

What Is the Oversubscription Ratio?

This is the ratio of:

- **Downlinks** (to servers/storage)
- **Uplinks** (to the network core or spine)

Example: A 3:1 ratio means 3 servers share 1 uplink.

Modern Datacenter Rule

Today's data centers try to keep this ratio **at 3:1 or lower** to support fast East-West (server-to-server) traffic and reduce bottlenecks.

What Are Uplinks and Downlinks?

In a network switch:

- **Downlinks** go **to the servers or storage** (lower layer)
- **Uplinks** go **up to the spine/core switches** (higher layer)

It's easier to see in a picture, but think of it like an elevator:

- Going **down** = talking to servers
- Going **up** = sending data to the network core

Why the Ratio Matters

The **uplink/downlink ratio** affects **how much bandwidth** servers can use.

Too few uplinks = bottlenecks (slow network performance).

Recommended Ratio

A common goal is:

- **1/3 bandwidth for uplinks**
- **2/3 bandwidth for downlinks**

This helps avoid congestion while using resources efficiently.

Example Switch Setup

Suppose you have a switch with:

- **20 QSFP ports** (each 40 Gbit/s) → for **downlink**
- **4 QSFP28 ports** (each 100 Gbit/s) → for **uplink**

Total Bandwidth:

- **Downlink**: 20 × 40 = **800 Gbit/s**

- **Uplink**: 4 × 100 = **400 Gbit/s**

This gives you a **2:1 downlink-to-uplink ratio**, which is pretty good for many use cases.

How to Increase Bandwidth in a Data Center

If the network becomes too slow because the **uplink bandwidth** (connections going up to the spine switches) is not enough, there are a few ways to fix it:

---

1. Add More Spine Switches

- This gives **more paths** between the leaf switches and the core.
- Helps balance traffic better (using protocols like ECMP).
- Lowers the **oversubscription ratio** because there are more uplinks.

---

2. Add More Leaf Switches

- Spread the servers across more leaf switches.
- Each switch handles fewer servers = more bandwidth per server.
- **But beware**: the spine switches must be powerful enough to connect to the new leaf switches.

---

3. Add a Super-Spine Layer

- This is for **very large data centers**.
- A "super-spine" is added **above the spine layer** to handle even more connections.
- This does increase the number of hops (slightly more delay), but it allows support for **many more servers**.

Realistic Example with Calculations

Let’s see what the numbers might look like in a real setup:

- **LACP** (Link Aggregation Control Protocol) can group up to 8 ports between a spine and a leaf, but that’s rare due to space and cost.

Now let’s assume:

- Each **spine switch** connects to **20 leaf switches**
- Each **leaf switch** has **48 downlink ports**, grouped in sets of 4 → 12 servers per leaf
- You have **6 spine switches**

So the total:

- **20 leaves × 12 servers = 240 servers**
- That’s a large capacity — for reference, **UniPi’s datacenter** has only **62 physical servers**.

Even if you reduce the setup (say 3 spines instead of 6), you'd still support around **120 servers** — more than enough for many real-life cases.

How Does a Leaf Switch Choose the Best Path?

In a **spine-leaf network**, each leaf switch is connected to all the spine switches. So when it needs to send data, it has **multiple paths** it can choose from.

ECMP – Equal-Cost Multipath Routing

The leaf switch doesn’t manually “know” which path is best. Instead, it uses a smart routing method called **ECMP**(Equal-Cost Multipath).

- **ECMP spreads traffic** across all paths that have the same cost (same number of hops).
- This way, no single path gets overloaded — traffic is **automatically balanced**.
- If one path becomes congested, ECMP can **shift traffic** to a better path.

Does the Leaf Know About Other Leaves?

No — a leaf switch doesn’t need to know what other switches are doing.

- It just **sends traffic using ECMP**, which makes decisions based on network rules and sometimes basic statistics (like hash-based traffic balancing).

- The **network does the optimization**, not the individual switch.

Why This Works Well

This system ensures:

- **High performance** even during heavy traffic
- **Efficient use of all available bandwidth**
- **No need for complex coordination between switches**

What Is a Full Fat Tree?

A **full fat tree** is a special kind of **network design** used in big computing systems (like supercomputers or High Performance Computing – HPC). It's based on the **spine-leaf architecture**, but with **more bandwidth** in the upper parts of the network.

Why "Fat" Tree?

Think of it like a real tree:

- The **top branches** (closer to the core of the network) are **thicker** because they carry **more traffic**.
- The **lower branches** (closer to servers) are **thinner**, because they serve fewer connections.

So, "fat" means **more capacity (bandwidth)** where it's needed most.

What Problem Does It Solve?

The **full fat tree** solves the problem of **oversubscription**, where:

- Too many servers share too little uplink capacity.
- This can slow things down when many devices send data at the same time.

With a **full fat tree**, the **oversubscription ratio is 1:1**, meaning:

- The **total uplink bandwidth** = **total downlink bandwidth**.
- No bottlenecks. Every server can use its full speed all the time.

When Is It Used?

- Mostly used in **supercomputers** or **data centers** that need maximum performance.
- Rare in typical networks because it's **expensive** (requires a lot of high-speed links and switches).

Key Difference: Layer 2 vs Layer 3

There is no routing in layer 2, broadcast is the standard way of communicating. Potentially anyone who is physically in the same LAN can see the traffic of almost anyone else, slightly depending on the fabric.

VLANs: The Solution at Layer 2

With VLAN frames are extended by 4 bytes. Every switch nowdays automatically sets the VLAN ID to 1; if the field is not existent, it is appended, making an untagged a tagged frame.

Switches ensure that data cannot spill/leak from a VLAN to another, because they avoid sending traffic on ports which are not associated to the current VLAN.

VLAN became largely of use when 10Gbit connection came out, because only 1Gbit was a too constrained bandwidth to be splitted into multiple VLANs.

VLAN are used to partition the traffic at data link layer without having to redo the fabric. They are particularly useful in cloud environments.

VLAN Port Modes (Cisco Terminology)

1. **Access Mode**
   - Connects to end devices like PCs.
   - Carries traffic for only one VLAN.
2. **Trunk Mode**
   - Connects switches or VLAN-aware devices.
   - Carries multiple VLANs tagged on the same port.
3. **General Mode**

- Can carry multiple untagged and tagged VLANs on the same port.
- Only one VLAN is set as the native (PVID), which handles untagged incoming traffic.

Why VLANs Are Important?

- They let you **partition traffic without changing physical wiring**.
- Vital for modern data centers and cloud environments, where many virtual networks coexist on the same hardware.

Two Parts of a Switch

Control Plane

This is the part of the switch that tells the other part (the data plane) what to do.

It runs an operating system (OS). In the past, this OS was usually locked and only worked in specific ways. Now, most switches use more flexible or even open (free to modify) operating systems. **Example:** Dell switches now use an open OS.

Data Plane

This part does the actual work with the data. It sends and receives data, runs network rules (protocols), handles VLANs, etc. It uses a special chip to do these jobs quickly and efficiently.

Connecting the Two Planes

The control and data planes are connected by a slower link (PCIe).

This is enough because the control plane doesn't need to send a lot of information.

OpenFlow

OpenFlow is a tool that lets you control what the switch does with the data.

It manages the "flow table," which contains the rules for handling data.

It helps the control plane and data plane work together.

It is possible to use a very fast and simple —reduced number of keystroke down to the strict necessary ones (e.g. en instead of enable)— CLI to program a switch. It is also possible to create a script file to be automatically executed by the switch at boot time.

Storage

Data Loss - “Storage is crucial because, if a switch fails, or a server fails, the service will be interrupted, but the data will still be there. If the storage fails, the data will be lost.”
Data is the most important of a system. Since data loss is permanent, the storage is completely different from computing or networking.

Storage Speed Then and Now

In the past, storage was the slowest part of the computer.

It worked in milliseconds (ms), while the CPU worked in nanoseconds (ns).

Now, thanks to SSDs, storage is much faster — working in microseconds (µs).

That’s about **100 times faster** than old hard drives.

What is NVMe?

NVMe stands for *Non-Volatile Memory Express*.

It is **not** a physical part — it's a **protocol**, like a set of rules.

It lets storage talk directly to the computer through the **PCIe bus**.

This means it doesn’t need to go through the slower **SATA controller**.

Why NVMe is Better

- **Faster data transfer (higher throughput)**
- **Lower delay (latency)** This makes computers much quicker when reading or writing data.

Why a 15TB Disk Could Be Better Than a 27TB Disk

(Assuming same speed and price)

Faster to Read Everything

If the disk is smaller (15TB vs. 27TB), it takes **less time to read all the data** from it. So, if you ever need to move or copy everything, the 15TB disk is faster.

When Bigger Disks Make Sense

Larger disks (like 27TB) are used for **cold storage** — this means:

- Data is stored for a long time
- It's not accessed often
- Only small parts are read at a time
- Speed is **not important**

Even if the disk fails and you need to recover everything, it's usually **not urgent**, since failures are rare.

Summary

- **15TB is better** if you care about speed when copying all the data
- **27TB is better** for storing lots of data that you rarely use

SSDs

- Invented by **Toshiba in 1980**
- Not widely used until about **30 years later**, when they became affordable
- Some large SSDs (e.g. 30TB) only use part of the space (like 10TB) for storing data — the rest is used for **redundancy**, which helps the disk last longer (up to 3× more)

What are TLC and QLC?

- **TLC = Triple Level Cell** → stores **3 bits per cell**

- **QLC = Quad Level Cell** → stores **4 bits per cell**

More bits per cell means:

- **Lower cost**
- **Slower performance**
- **Less reliable storage** (harder to read/write and to keep data safe)

When to Use TLC vs. QLC

- **TLC**
  - Used for **hot storage** (data accessed often)
  - **Faster**, **more reliable**, **longer lifespan**
  - **More expensive**
- **QLC**
  - Used for **cold storage** (data accessed rarely)
  - **Cheaper**, but **slower** and **less durable**

What is Tiering?

Tiering = Smart way of storing data based on how often it's used

- **Frequently used data** → stored in **fast, expensive storage (like TLC SSDs)**
- **Rarely used data** → stored in **slower, cheaper storage (like QLC SSDs)**

This helps:

✅ Improve speed and performance

✅ Lower overall storage cost

What is IOPS?

IOPS = *Input/Output Operations Per Second*

It's a number that shows **how fast** a storage device (like an SSD) can handle read/write operations.

IOPS performance depends on **how the data is accessed** either **randomly** or **sequentially**.

Random vs. Sequential Access

- **Sequential Access**
  - Data is read or written in order (one block after another)
  - Best for frequent reads of the full files are required
- **Random Access**
  - Data is read from or written to **different locations**
  - Best for frequent direct access and working with **specific records** or small updates

What Are Queues in I/O?

- Each program (or thread) can have its own **queue** of I/O requests
- The OS manages these queues to handle requests **asynchronously** (in the background)
- **Multiple queues** help increase **throughput** (more tasks finished per second)
- If a task is **very sensitive to delay (low latency)**, it's better **not to use a queue** — so the task can run immediately without waiting

What is Latency in Storage?

Latency = the **delay** before data starts being read or written.

- On a **mechanical hard drive**, reading 40MB of data adds about **2.71% latency**
- **Optane** (a fast storage tech) can do **416 operations** in the time an HDD does **just 1**

  → It feels like latency is **almost zero**

Because of this, it may seem like doing many small read/write operations is "free" — but that's not always true.

Where Does Latency Come From?

1. **Software latency** (µs level)
    - Caused by the operating system and cannot be removed
2. **Controller latency**
    - Improved with **NVMe** as low as **20µs**
3. **HDD latency**
    - Very high in old hard drives
    - Greatly reduced with **SSDs** and even more with **3D NAND**

What is Storage Aggregation?

Storage aggregation = combining **multiple storage devices** into **one big virtual drive**

This helps by:

✅ **Increasing performance** (many read/write tasks at once)

✅ **Reducing latency** (requests are handled in parallel)

✅ **Improving reliability** (data can be duplicated across disks)

Even if your data is spread out, the system sees it as **one big drive**, which makes it faster and smarter.

What is Fibre Channel?

Fibre Channel is a special type of fabric **used only for storage**.

It connects storage devices (like SANs) to servers.

How It Works

- The storage device connects to the server through a **HBA (Host Bus Adapter)**
- The HBA makes the remote storage look like a **local drive** to the server
- This allows fast and reliable access to the storage

Connection Types

- Fibre Channel usually uses **optical fiber cables** (fast and high-capacity)

- But it can also run over regular **Ethernet cables** — this is called **FCoE** (Fibre Channel over Ethernet)

What Is a Bus?

A **bus** is a communication system that lets **multiple devices connect and talk** to each other.

It has:

- A **clock** to keep everything in sync
- Several **lanes** (data paths) — for example, **PCIe has 16 lanes**

Each PCIe lane gives around **1 GB/s** of bandwidth.

So a full 16-lane PCIe bus = **~15 GB/s total**.

How NVMe Uses the Bus

- A single **NVMe SSD** can transfer data at around **3.5 GB/s**
- **4 NVMe SSDs** × 3.5 GB/s ≈ **14 GB/s** → this is enough to **fully use (saturate)** a PCIe bus
- The same 4 drives also **fill up a 100 Gbps network** (which is 12.5 GB/s)

NVMe and RAM

- NVMe is **slower than RAM**, but not by much — just **one order of magnitude** slower
- It offers **much higher capacity** than RAM
- Because of this, NVMe is now used like a **super-fast cache** or **extra RAM**
- Some systems combine RAM and NVMe into one big memory space, **in a way that's invisible to software**

Where Is the Bottleneck Now?

- **Disk latency** is around **5 microseconds** (very low)
- But **TCP/IP network latency** is around **70–80 microseconds**

→ The **network** is now the slowest part, not the storage

→ Also, **network bandwidth** is a limit: just **4 NVMe drives** can **saturate** a 100 Gbps connection

What Are Checkpoints?

- A **checkpoint** is a saved state of the system at a specific point in time
- If something goes wrong (e.g. crash), the system can be **restored to the last checkpoint**
- Any data written **after** the checkpoint can be **re-applied**, like a fast replay

---

Why Checkpoints Matter

- It's not practical for a system to lose **months of work**
- Checkpoints help reduce **data loss** and make recovery **faster and easier**

What Is RAID?

RAID = Redundant Array of Independent Disks

(A long time ago: "Inexpensive" Disks, when disks failed more often)

RAID combines **multiple disks into one virtual drive** to:

✅ Increase **reliability** (fault tolerance)

✅ Improve **performance** (faster reads/writes)

✅ Or do both

- **RAID 1 =** two identical drives, when reading you may even get faster speed than a single drive. **Mirror** content of two drives. Duplicates data to two disks (mirror).
  - **CONS =** pay two disks to have half the space
- **RAID 5 =** almost not used by anyone. **XOR operator** = if delete one of the xor columns you can recover the missing information by doing xor of elements.

Use XOR operator, split data in two, write part of data in 1st drive, part in 2nd and in 3rd drive put xor data of the two. If you lose one drive you can reconstruct all bits of that drive by using the bits of the other two drives, data is **not accessible** during reconstruction of the data (not in RAID1, when you insert a new drive the reconstruction takes place) Stripes data + **parity (error correction)** across 3+ disks. One isk can fail

- **RAID 0 = NO REDUNDANCY (STRIPING = split data across two or more disks)**, it says that if you have multiple drives you can address it as a single drive to pile up the space. Like joining two drives as a single drive. Fast, if one disk fails data is lost.
- **RAID 6 =** similar to raid 5 but I can lose 2 drives, pay as much as RAID 1 (need two times the written space). Can stay aligned even if you lose two drives, is not a generalization of raid 1.

Network Sharing Architectures

Protocols vs Architectures

Before understanding **architectures**, we need to know what **protocols** are.

- **Protocols** are **rules** that define how data is shared over the network
- **Architectures** describe **how the systems are built and connected**

So, **SMB** and **NFS** are **protocols**, **not** architectures.

File Sharing Protocols

SMB / CIFS

- **Used by:** Windows (but also works on Linux and MacOS)
- **Strength:** More **secure**
- **Weakness: Slower** than NFS

NFS (Network File System)

- **Used by:** Linux and MacOS (but also works on Windows)
- **Strength: Faster**

- **Weakness:** Less **secure**

Capacity and system architecture

When we talk about capacity, there are two measures which we can refer to:

1. **Scale-up:** adding more disks to the same server
2. **Scale-out:** adding more servers to the same network

File-Based Storage – NAS

What Is NAS?

NAS = Network Attached Storage

It's a **device connected to the network** that stores files and makes them available to **multiple users and devices**.

Key Features:

- Acts like a **shared folder** over the network
- Has its own **CPU, RAM, NICs**, and **optimized OS** for file serving
- Uses **file-sharing protocols** like:
  - **SMB/CIFS** (Windows)
  - **NFS** (Linux/macOS)
  - **AFP** (macOS)

Use Case:

- Great for **file sharing**, **document storage**, and **collaboration** across teams
- Easy to use and manage, just like accessing a folder over Wi-Fi or Ethernet

Block-Based Storage – SAN

What Is SAN?

SAN = Storage Area Network

It connects storage devices to servers so that they look like **local disks** to the serve r's OS.

Key Features:

- Shares **blocks of data**, not files
- Uses **HBAs (Host Bus Adapters)** to connect servers to storage
- Runs on its own **separate network** (not on the LAN), using:
  - **Fibre Channel** (fast)
  - **iSCSI** (slower)

Smart Features:

- Supports **deduplication**: removes duplicate blocks to save space
- Can improve performance by splitting data across multiple drives and reading in **parallel**

SAN Architecture:

- Traditionally has a **"head"** (controller) managing drives and connecting to servers
- The head is connected to a **Fibre Channel switch**, and then to storage drives
- Servers access storage **as if it's local**, thanks to the head

---

The SSD Problem:

- SSDs are **too fast** for traditional SAN heads
- Example: 4 SSDs can **saturate** a 10 Gbit link
- Solution: **remove the head**, connect drives **directly** to servers → this is called **DAS**

DAS – Direct Attached Storage

What Is DAS?

DAS = Direct Attached Storage

The storage is **directly connected to one server** without going through a network.

Key Points:

- Uses **SCSI** protocol
- **Faster** than traditional SAN in some cases (especially with SSDs)
- **Not scalable** like SAN — each DAS is only for one server

---

Mechanical Drives Still Useful?

Yes!

- A group of mechanical drives, used smartly (e.g. data split across them), can still **outperform a single SSD**
- That's because **parallel access** across multiple drives boosts performance

Key Difference Between NAS and SAN

- **NAS** provides both **storage** and a **file system**, so it manages files directly (like a shared folder).
- **SAN** only provides **block-level storage**. The **client** (the computer or server using it) is responsible for managing the file system.

🔷 **Note:** A NAS device can also be included inside a **SAN network**, combining the benefits of both systems.

Storage Pools and LUNs

Storage Pools

- A **storage pool** groups multiple storage devices into one **big logical unit**.
- Purpose: improve **performance** and **reliability** (e.g. if one disk fails, others take over).

LUNs (Logical Unit Numbers)

- SAN divides storage into **LUNs**, which are like **virtual partitions** of the storage pool.
- A **LUN** looks like a real physical disk to a server.
- LUNs are used to:
  - **Assign storage** to servers.
  - Control **access using ACLs** (Access Control Lists).

Features of LUNs

🔄 1. **Dynamic Size (Virtual Provisioning)**

- You can **present a LUN** as larger than it actually is.
- The size can be expanded later when more space is needed.
- This helps:
  - Reduce **fragmentation**.
  - Avoid wasting unused space.

🧠 Example: Present a 2TB LUN, but only use 1TB at first. Expand later if needed.

📉 2. **Deduplication**

- **Removes duplicate blocks** to save space.
- Especially useful in environments where files (like documents) are often copied.

📦 3. **Compression**

- Stores data in a smaller form to save space and bandwidth.
- Can be:
  - **Lossless**: No data lost.
  - **Lossy**: Some data may be lost.
- Downside: It needs **CPU power** to compress/decompress.
- Upside: Saves **network bandwidth** when transferring data.

- Tools like **FM-index** allow searching inside compressed data.

📸 4. **Snapshots**

- Snapshots are **“point-in-time” copies** of the data.
- Useful to **restore data after a failure or accidental change**.
- Only the **differences** from the last snapshot are saved.
- Snapshots use disk space, and **old ones (e.g. >1 week)** are usually deleted to save space.

LUN Creation Methods

LUNs can be created in two ways:

- **From a RAID Set (Traditional)**
  - Best when you need **predictable performance**.
- **From a Storage Pool (Modern)**
  - Best when you need **scalability** and **flexibility**, and can accept some **performance variation**.

📏 Types of Capacity

- **Raw Capacity**:

  Total size from all physical disks combined.

- **Usable Capacity**:

  Space actually available for use, after subtracting metadata and system overhead.

- **Provisioned Capacity**:

  Space "presented" to the server.

  Can be **larger** than usable capacity → this is called **overprovisioning**.

Object-Based Storage – S3

- **S3** = *Simple Storage Service*
- Stores data as **objects** (not files or blocks).
- Each object includes:
  - **Data + Metadata** (size, owner, date, etc.)
  - **Attributes** (like retention, access info)
- Uses a **flat address space** (no folders) → easy to **scale**
- Often used for **cloud storage** and **Storage-as-a-Service**

Synchronization Software & Pricing

The “storage guy” must ensure that there is no condition under which can happen data loss, because it is never an option. It is also important to have powerful synchronization algorithms, which must allow data to be copied and synchronized in multiple locations without disrupting performance and handling concurrency; such software is typically costful.

It is difficult nowdays to establish what is the “right” price for software. The shift from highly specialized and costful hardware to general hardware-plus-software, gave the software, which still is a non-physical entity, increasingly more value, perhaps even too much.

Hyperconverged Infrastructure (HCI)

SAN started to create a sensible bottleneck, so designers started to “move drives towards the servers”. DAS stands for Direct Attached Storage, and is a technology that allows to connect multiple storage devices to a single server, in order to increase the performance, the reliability, or both. The limitations is that you can only attach up to 2 or 3 drives to a server.

An idea came out to use the servers’ internal drive to build a Storage Area Network, and this is called VSAN.

 HCI (Hyperconverged Infrastructure)

HCI stands for Hyperconverged Infrastructure, and is a technology that allows to combine multiple servers into a single logical unit, in order to increase the performance, the reliability, or both. The idea was born to allow a scale-out architecture, where you can add more servers to the network.

The **Hypervisor** is the software that allows to run multiple virtual machines on a single server. There should be some locality between the VM and the storage, because the VM should be able to access the storage quickly.
The controller VM (one per host) implements the storage abstraction and the logical moving of data.

- Built for **scale-out**: *"Add a server, get more capacity."*
- Each server runs a **Controller VM**:
  - Reads: Done **locally**.
  - Writes: Require **remote sync + acknowledgment** for data replication.

📡 Integration

- **New servers** are auto-integrated into the HCI cluster.
- Their **storage is added** to the pool automatically.
- **Local replicas** are created for redundancy.

🔄 VM Live Migration

- **Live Migration** = Move a VM *without downtime*.
- If storage is **shared** (as in SAN or HCI), there's **no need to move data**.
- In HCI, **local copies** may need syncing during migration.

🗃️ Riak and Acropolis

Riak is a distributed database that is used to store data in multiple locations. The same applies to Acropolis, which is a distributed storage system.

💾 SDS – Software Defined Storage

SDS Software Defined Storage refers to software for policy-based provisioning and management of data storage independently from the underlying hardware.

Such software is more costful than the hardware it is running on, since it also optizimes the drives, not simply managing them.

SDS exploits object-based storage architecture and DHTs to provide storage services.

Computing

One important notion of compute system is density: there are different needs (in how many cores / how many operations per second) depending on the application that is running on the system. The structure of a server is a tradeoff between capacity (even if we need to do a lot of computing, space is limited, so less space for the drives) and density.

There are two types of compute systems in a datacenter:

Rack servers: they are the most common type of servers, and are used for general purpose applications.
Blade servers: they are used for specific applications, and are more expensive than rack servers. Each blade server is a self-contained compute system, inserted in a —fatter— chassis, typically dedicated to a single application, providing multiple services.
The modular design of blade server make them smaller, minimizing floor space requirements, increasing system density and scalability, ultimately providing a more efficient use of power and cooling, compared to tower and rack servers.
Usually contain only CPU(/RAM) and NIC, but in some cases also storage, possibly configured as SAN or NAS.

Remote Management of servers

“In a server how many OSs are ran at the same time?”
2 in general, one is “Base Management Console”1.

The BMC is a full OS running in a board which executes also when the server is off, but attached to a power supply.
It is a component which allows you to manage the server as if you were phyisically handling the server.
Prof. Cisternino display iDRAC (Dell's BMC) in class. It is a web interface which allows you to manage the server,
check its status, and even turn it on and off.

It is also possible to access a console, which is a virtual console, which allows you to interact with the server as if you
were physically there with keyboard and mouse attached; such console also allows to install a new OS by uploading
an iso and make the server boot from it as if it was attached to it.

The BMC typically has a dedicated network interface, which is used to connect to the server, separated from the standard network interface. Redfish is a standard which is used to manage servers. It is a RESTful API which allows to manage servers and automate some tasks.

The BMC is very useful because allows for administrators to manage server remotely, without having to physically access the server. This is not only "handy", but often necessary, since the physical security of the datacenter must be high, and not everyone should be allowed to access the server room.

Measuring Bus Speed
PCIe speed, as well as CPU speed is measured in GT/s, which stands for Giga Transfers per second. It is a measure of how many transfers can be performed in a second.

Predict Storage failure

A disk may fail without any prior notice, at any time, just like a heart attack.
However there are some signs which may indicate that a disk is about to fail; these are detected by the Smart technology. Also AI may be used to predict disk failure.

Knights Landing (KNL) – Intel Architecture for HPC

Knights Landing is an old multi —up to 72— core architecture designed to be used in supercomputers. The memory had a super high bandwidth, to avoid

bottlenecking the many cores inside, since such memory is shared among them.

NUMA is a technique used to avoid bottlenecking in multi core architectures. It is a technique which allows to have multiple memory banks, each one connected to a subset of the cores. This way, each core can access its own memory bank without having to wait for the others to finish accessing the shared memory.

Rings and CPU Architecture

Bachelor's professors fooled us into thinking that CPUs have two operating modes, user and kernel mode. Sadly, this ain't true, it is an abstraction. In reality, *CPUs have 4 rings*, which are used to separate the different levels of privilege. The higher the ring, the higher the privilege level. The kernel runs in ring 0, while the user runs in ring 3.

Nowadays there are multiple units in the CPU, which are used to execute instructions, and there is a head unit which decides which instruction to execute next and on which unit.

Chiplets are a way to increase the number of cores in a CPU. They are small chips which are connected to the main CPU. Even at CPU the "general-purpose" methodology is not feasible anymore, and the CPU is divided into multiple units, each one specialized in a specific task.

Chiplets

- **Chiplets** are a design innovation where CPUs are built from **multiple small dies** instead of a single large one.
  - They help with **manufacturing yield**, **cost**, and **scalability**.
  - Each chiplet can be specialized (e.g., compute, cache, I/O), allowing better **modularity** and **reuse**.
- This reflects a broader industry shift: **"general-purpose CPUs" are fading** in favor of **specialized compute blocks**.

Misc Notions on Hardware

GPUs became of paramount importance for datacenter in the last years, mostly because of the rise of AI and machine learning. They are used to perform parallel

computations, and are much faster than CPUs for such tasks. However, they are very expensive.

NPUs (Neural Processing Units) are a new kind of processors, which hardware acceleration for for AI and machine learning tasks. They are much cheaper than GPUs, and are becoming more and more popular.

Virtualization

Virtualization means using software to create virtual versions of physical resources like CPU, memory, storage, and network. This lets you run **many operating systems (OSs)** on one physical computer (called the **host**).

This is **not the same as emulation**:

- **Emulation** copies how another machine works, but it's **very slow**, because each instruction may need to be translated into many steps.
- **Virtualization** uses the real hardware more directly, so it's **much faster**.

CPU Rings and isolation

Virtualization is a strong way of isolating things.
To isolate VMs the hypervisor exploits CPU rings, which are used to separate the different levels of privilege. The higher the ring, the higher the privilege level. In intel CPUs there are typically 4, but up to 16 rings.

Two Types of Virtualization:

1. **Desktop Virtualization**

   Used for personal or graphical use (with mouse, screen, etc.):

   - Examples: **VirtualPC, VirtualBox, VMware Workstation, Parallels**
   - Goal: Run another OS (like Linux on Windows) with a full desktop.

2. **Server Virtualization (Hypervisors)**

Used in datacenters for running multiple **headless (no GUI)** servers:

- Examples: **VMware ESXi, Hyper-V, KVM, Xen**
- Goal: Use one machine to run many virtual servers efficiently.

Extra Features:

- **Virtual Switch**:
  - Software that connects virtual machines to the network.
  - The "uplink" of this switch is the host's actual network card.
  - Like a real network switch, but virtual.
- **Checkpoints (or snapshots)**:
  - Save the VM's state (like a "save game").
  - Useful for testing – if something goes wrong, you can go back.

Network in Virtualization

When using virtualization, **Hypervisors** (like VMware ESXi, Hyper-V, etc.) include a key tool called a **Virtual Switch**(vSwitch).

- A **vSwitch** is software that acts like a network switch for virtual machines (VMs).
- Each VM has its own **virtual network card** and **MAC address**, just like a real machine.
- These virtual cards connect to the **vSwitch**, letting the VM talk to other machines or the internet.
- One hypervisor can manage **multiple vSwitches**.
- If the vSwitch or host network is in **promiscuous mode**, it can see all the traffic passing through it — this might include the traffic of other VMs too.

VMware is a leading virtualization company, but some users are unhappy due to **price and licensing changes**.

Broadcom, a chip company, now owns VMware and might build virtualization directly into **hardware** (future possibility).

Live Migration (Moving VMs Between Servers)

Live Migration means moving a virtual machine (VM) from one physical host to another **without turning it off**.

- In **VMware**, it's called **vMotion**.
- In **Microsoft Hyper-V**, it's just called **Live Migration**.
- This is useful for:
  - **Maintenance** (e.g., update hardware or software)
  - **Load balancing** (moving VMs to reduce load on a busy server)
  - **Upgrades** (move to newer versions with no downtime)

How Live Migration Works (Simple Steps):

1. The **RAM** and CPU state of the VM is copied to the new server.
2. The VM **starts running** on the new host.
3. While it's still being used, more memory is copied as needed.
4. Once everything is fully moved, the **old VM is shut down**, and the new one takes over.
5. The **network settings** (vSwitch, MAC address, etc.) are also updated, so nothing breaks.
   - If needed, the **old vSwitch** helps forward network traffic to the new VM until the **network tables (ARP)** are updated.

More Detailed Steps (Advanced But Simplified):

1. Start copying the **RAM** while the VM is still running.
2. Create an **empty virtual disk** on the new host.
3. Quickly copy the **CPU state** (this is when the VM pauses briefly).
4. Make sure the **vSwitch knows about the move**, and if needed, forward traffic.

5. If the VM's storage (disk) is **not shared**, copy the disk data now using **jumbo frames** (large packets) to avoid slowdowns.

> 🔷 Note: If both hosts use the same shared storage, there is no need to copy the disk, which makes the process faster.

Replication is the process of copying data from one host to another, in order to have a backup in case of failure. Happens the same way as live migration, but the virtual machine is not running on the other host.

Containers

Why Use a Virtual Machine (VM)?

- A **VM is safer than a normal process** because it is **isolated**.
- If a hacker breaks a **normal process**, they can often access the whole system.
- If a hacker breaks a **VM**, they're **stuck inside the VM**, not the real machine (the host).
- But: VMs are **slower** because they need to run:
  - a full operating system,
  - a **hypervisor**, and
  - extra layers like virtual storage, memory, etc.

Containers (Like Docker)

- A **container** is like a **lighter alternative** to a VM.
- It's just a **process**, but it **thinks** it's running in its own system.

- The container only sees **part of the host's filesystem** and cannot affect the rest.

How It Works:

- The container's root folder is **mapped** to a **subfolder of the host**.
- Docker uses a **differential filesystem** — it only saves the **changes** from the original system.
- This makes containers:
  - **fast**,
  - **small**, and
  - easy to move and copy.

Dockerfile:

- A **Dockerfile** is like a **recipe**.
- It defines what goes into the container — like:
  - which operating system image,
  - what files to add,
  - what commands to run.
- You can use **temporary containers** during the build, then combine them into a **final clean version**.

VMs vs Containers: Key Differences

Feature	Virtual Machine	Docker Container
Isolation	Full system	Process-level
Performance	Slower (more overhead)	Faster
Size	Big (includes full OS)	Small (just the app & dependencies)
Network	Has its **own MAC address** via vSwitch	**Shares host MAC address**
Use Case	Full system emulation	Lightweight app execution

How to Safely Update a System (Upgrade Steps)

1. **Take a snapshot** (like a photo of the current state).
2. **Stop the containers**.
3. **Update them** (especially databases carefully).
4. **Restart the containers**.
5. **Check everything works**.
6. **Delete the snapshot** if all is okay.
7. If there's a problem, **roll back** using the snapshot.

Docker Compose & Kubernetes

- **Docker Compose**: Tool to manage **multiple containers at once**.
  - Uses a **YAML file** to define:
    - which containers to run,
    - how they connect (networks),
    - and shared data (volumes).
  - Start everything with **one command**.
- **Kubernetes**: A more **powerful tool** to manage **many containers**.
  - Good for **large-scale systems**.
  - Can **automatically scale**, heal, and manage thousands of containers.

Docker & Security

- Containers are **pretty secure**, but not perfect.
- One possible attack: a hacker uses the container to **stress the system** and then **observe performance drops**.
  - This may leak info about what other apps are doing.
  - This is called a **side-channel attack**.

Cloud

What is the Cloud?

The **Cloud** is not just a technology — it's a **business model**. Companies started using it to handle **peaks of demand** without buying too much expensive equipment.

Example:

Amazon gets tons of website traffic during **Christmas**, but not so much on regular days. Instead of buying many servers just for Christmas (and letting them sit unused all year), they use **cloud services** that can grow or shrink when needed.

Key Cloud Concepts

1. **Scalability & Cost Saving**

   You don’t need to buy big servers. You can “rent” computing power when needed, saving money.

2. **Resource Pooling**

   Cloud providers (like AWS, Azure, Google Cloud) share their servers across many customers. You **don’t know exactly where your program runs or where your data is stored**, but the system handles it for you.

3. **Elasticity**

   The cloud can automatically **add or remove resources** (CPU, RAM, storage) based on how much is needed.

   → To the user, it feels like there are **unlimited resources** available.

4. **Usage Tracking (Metering)**

   Cloud providers measure how much you use (like electricity), so you **pay only for what you use**.

Benefits of Cloud

- **Fast and Flexible**

  You can quickly start new services or scale them. It helps you build and release apps faster.

- **Lower Costs**
  - You don’t need to buy hardware upfront (low CAPEX).
  - You use resources more efficiently.
  - Maintenance costs go down (low OPEX).
- **High Availability**

  Systems are designed to always be available — even if one part fails.

  Example: **Active-active** or **active-passive** setups.

- **Business Continuity**

  If something breaks, the cloud can switch to backup services with no downtime.

- **Scaling and Access**
  - Easily grow or shrink resources.
  - Access from anywhere.
- **Better Collaboration**

  Multiple teams can work together more easily in the cloud.

- **Simpler Management**

  The cloud hides the complicated stuff (hardware, networking, etc.) and makes things easier to manage.

- **Separation of Hardware & Software**

  You don’t have to worry about the physical machines — just focus on your app or service.

Downsides of Cloud

- **Vendor Lock-In**

  Hard to switch providers once you start using one (like AWS or Azure).

- **Privacy Risks**

  Your data lives on someone else's servers.

- **You Depend on Someone Else**

  If your provider has issues, your app might go down too.

- **Legal Problems**

  Some laws (especially in the EU) say your data must be stored in specific locations (e.g., within Europe).

- **Tricky SLAs**

  Service Level Agreements (SLAs) may look good, but small print can limit what's guaranteed.

Cloud Service Models (Simplified)

There are 3 main types of cloud services:

---

1. IaaS – *Infrastructure as a Service*

➡️ The cloud gives you **basic computer stuff** like virtual machines, storage, and networks.

You manage the operating system and applications yourself.

Think of it like renting a server.

Example:

- Amazon Web Services (AWS EC2)
- Google Compute Engine

---

2. PaaS – *Platform as a Service*

➡️ The cloud gives you **a ready-to-use platform** to build and run your apps.

You don't worry about the servers or operating system — just focus on writing code.

Example:

- Google App Engine

- Microsoft Azure
- Heroku

---

3. SaaS – *Software as a Service*

➡️ The cloud gives you **ready-to-use software** through a website or app.

You just log in and use it — no need to install or maintain anything.

Example:

- Google Docs / Gmail
- Microsoft Office 365
- Salesforce

Cloud Deployment Models (Made Easy)

These are the **different ways** cloud services can be set up and shared:

---

1. Public Cloud

➡️ Owned and managed by companies like Amazon, Google, or Microsoft.

➡️ Anyone can use it — businesses, schools, individuals.

Important:

"Public" **does not mean** your data is visible to everyone.

It just means **anyone can rent the service**.

Examples: AWS, Microsoft Azure, Google Cloud

---

2. Private Cloud

➡️ Used **only by one organization** (e.g. a company or a university).

➡️ Can be managed by the company itself or another provider.

➡️ Can be **on-site** (inside the company) or **off-site** (in another location).

Important:

"Private" **does not mean** your data is fully protected by default — it means **only one group** uses that cloud.

Example: A private cloud used by UniPi (University of Pisa)

---

3. Hybrid Cloud

➡️ A mix of **public + private clouds** working together.

➡️ You can move data and apps between them.

Example: A company stores sensitive data in a private cloud and uses public cloud for heavy processing.

⚠️ **Note:** Moving stuff between clouds can slow things down or cost more.

---

4. Community Cloud

➡️ Shared by **a group of organizations** with similar needs (like universities, banks, or hospitals).

➡️ Can be managed by one of them or a third-party.

➡️ Can be on-site or off-site.

Example: Multiple hospitals sharing a cloud system to manage patient data securely.

Physical Layer (Basic Level of the Cloud)

This is the layer that includes all the **real, physical equipment** — like servers, storage systems, and networking gear — used to **run the cloud**.

Think of this as the **"hardware base"** of the cloud, where everything actually happens.

What does it include?

1. **Compute**

   This is about the **processing power** — usually offered as **virtual machines (VMs)**.

   These are like computers running inside another big computer.

2. **Storage**

   Where the data is saved.

You've already studied things like:

- File-based storage (like folders and files)
- Unified storage (a mix of different storage types)

3. **Network**

   How all the cloud parts **talk to each other**.

Types of Network Communication

1. Compute-to-Compute (also called **East-West traffic**)

➡️ When two VMs or servers talk to each other.

➡️ They usually use **IP protocols** (standard internet communication).

2. Compute-to-Storage

➡️ When a computer accesses the storage system (to save or load data).

➡️ This is usually done using **SAN** (Storage Area Network), and there are a few different types:

- **FC SAN (Fibre Channel SAN)**

  Very fast, high-performance storage connection.

  Up to **16 Gbps** speed.

  Works at the **block-level** (low-level, fast access).

- **IP SAN**

  Uses normal internet/network cables (Ethernet).

  Runs **iSCSI** or **FCIP**, which wrap special storage data inside regular IP packets.

  Easier to set up than FC SAN.

- **FCoE (Fibre Channel over Ethernet)**

  Combines Fibre Channel (fast storage) with Ethernet (standard networking).

  So you can use one network for both data and storage.

Inter-Cloud Communication (Inner-cloud)

This is when **one cloud connects to another cloud**, often using the **WAN** (Wide Area Network), like the Internet.

Useful for hybrid or multi-cloud setups.

Virtual Layer (The Layer That Creates Virtual Machines)

The **virtual layer** is where **software turns physical hardware into virtual parts**.

It's like using a powerful computer to **create many fake computers** (called **Virtual Machines**, or VMs), so multiple systems can run at the same time, even though there's only one actual machine underneath.

Why Do We Need This Layer?

- It helps one machine act like **many machines** (multiple VMs on one server).
- It also helps many machines act like **one system** (combine power for one big task).
- You can **share resources** (CPU, memory, storage, network) better.

What Makes This Work? Hypervisors

Hypervisors are special programs that **create and manage virtual machines**.

There are **two types**:

1. **Bare-metal Hypervisor**
    - Installed **directly on the hardware** (no operating system in between)
    - Faster and better for big data centers
    - Example: VMware ESXi
2. **Hosted Hypervisor**
    - Runs **on top of a normal operating system** (like Windows or Linux)
    - Easier to use on personal computers, but slower
    - Example: VirtualBox

Network Components Inside Virtual Machines

1. **vSwitch (Virtual Switch)**
    - Acts like a network switch inside the virtual world (Layer 2 of OSI model).
    - Connects VMs to each other, or to the outside network.
    - Can be **internal** (only between VMs), or **external** (connected to physical network).

    📌 A physical network card (NIC) can only be linked to one vSwitch at a time.

---

1. **vNIC (Virtual Network Interface Card)**
    - It's the **virtual network card** inside each VM.
    - Each vNIC has its own **IP address and MAC address**, just like real network cards.
    - It connects the VM to a vSwitch.

---

1. **Uplink NIC**
    - A **real** (physical) network card connected to the **outside world**.
    - It **does not have its own IP** or MAC address seen by VMs.
    - It just forwards traffic **between** the VM network and the real network — like a **bridge**.
    - That's why it's called an "**uplink**": it links the virtual world to the physical one.

Virtual Networks

1. VLAN
2. PVLAN (Private VLAN)
3. Stretched VLAN
4. VXLAN (Virtual Extensible LAN)
5. VSAN

There exists a mapping between VSAN and VLAN to determine which VLAN carries a VSAN traffic.

Control Layer

The control layer is responsible for managing the resources and the allocation of the resources to the virtual machines.

(Control Layer) "The control layer is important because it's the way to poll the resources set and see all the resources in a coherent way so it's a sort of a software layer that hides the differences and gives you primitives (such as "I need a VM, I don't care where, but I need one")."

Note that you cannot allocate more virtual cores than the physical ones —same applies to memory—, but you can allocate smaller pieces so you can create a resource pooling of resources that can be taken partially (assuming that they can run on a single node), but making you perceive it as a pool of resources.

Service and Orchestration layers above the control layer have no clue where workloads are running, they just send
requests to the control one, it is completely up to such layer to manage where and how.

The steps towards provisioning a resource are three

1. Resource Discovery
2. Resource Pooling
3. Resource Provisioning

A key component in the control layer is the manager, which may be an Element Manager or a Unified Manager software, which handles, by means of APIs, the tools associated to

- Compute System management
- Fabric management
- Storage System management

Resource Discovery (Finding What's Available)

- The **control layer** (like a manager) checks what **hardware and software resources** are available.

- It **shows and organizes** all the resources (like CPU, memory, storage) in one place.
- This makes it easy to **see and manage everything** from a central dashboard.

Resource Pooling and Provisioning (Organizing and Giving Out Resources)

- The **control layer** groups resources into **pools**.

  Think of it like grouping similar items in baskets:

  - One basket has **fast SSDs**
  - Another has **large HDDs**
  - Another has **lots of RAM**
- These pools are **labeled** or **graded** like:
  - **Gold** (fastest, most expensive)
  - **Silver** (average)
  - **Bronze** (cheapest, slower)
- Customers can **choose** what kind of resources they want, without knowing exactly where the hardware is.

  Example: “I want 0.5TB Flash and 1TB of regular HDD” → the system finds the best fit from the pool.

Why is this important?

- It allows the cloud to offer **customized services**, without needing to match them to a specific machine.
- Makes it easy to **scale up or down** depending on customer needs.
- Helps with **cost control** — higher grade = higher cost.

Resource Monitoring

it is important to recall that Resource monitoring is fundamental to keep track of the resources used and to prevent over-provisioning.

Service Layer/Service Orchestration Layer

Cloud Service IT resources that are packaged by the service providers and are offered to the consumers

This means that deploying a service, does not mean simply deploying a VM, but a bunch of them, and also configuring it, installing software, etc.

Service Layer enables defining services in a service catalog, and provides a self-service portal for users to request
services (enables on-demand and self-provisioning).

Essentially, the catalog is the DB while the cloud portal is the web interface for it.

Service Orchestration Layer

Service Orchestration layer implements the process of integrating services to support the automation of business
processes: actuates the policies and the requests automatically from the user.

The slides report “Automated arrangement, coordination, and management of various system or component functions
in a cloud infrastructure to provide and manage cloud services.”
“Programmatically integrates and sequences various system functions into automated workflows”

Upon a service request by the customer, the orchestration layer triggers the appropriate workflows to provision the
service.

(Tenant) A tenant is a user of the cloud, and the cloud provider must ensure that the tenant is
isolated from the others. This is done by means of multi-tenancy.

Workflows

Although some manual steps (performed by cloud administrators) may be required while processing the service
provisioning and management functions, service providers are looking to automate these functions as much as
possible.

Cloud service providers typically deploy a purpose-designed orchestration software (“orchestrator”) that orchestrates the execution of various system

functions. The **orchestrator** programmatically integrates and sequences various system functions into automated workflows for executing higher-level service provisioning and management functions provided by the cloud portal.

The orchestration workflows are not only meant for fulfilling requests from consumers but also for administering cloud infrastructure, such as adding resources to a resource pool, handling service-related issues, scheduling a backup for a service, billing, and reporting.

We may generalize the lifecycle of a service as split in 4 main phases:

1. Planning
i. Assessing service requirements ii. Developing service enablement roadmap iii. Establishing billing policy

2. Creation
i. Defining service template ii. Creating orchestration workflow iii. Defining service offering iv. Creating service contract

3. Operation
i. Discovering service assets ii. Managing service operations

4. Termination i. Natural termination by contract agreement ii. Initiated termination by a provider or a consumer

Business Continuity

Business Continuity The capability of an organization to continue delivering products or services at acceptable predefined levels following a disruptive incident.

...or ...

BC entails preparing for, responding to, and recovering from service outage that adversely affects business operations

Business continuity involves *proactive* measures, such as business impact analysis, risk assessment, building resilient IT infrastructure, deploying data protection2 solutions (backup and replication). It also involves reactive counter-

measures, such as disaster recovery, to be invoked in the event of a service failure.

SPOFs and Remedies

SPOFs may occur at component level, or at site (data center) level.
The key to avoid SPOFs is to have redundancy, which may be achieved by means of replication or backups (more
on this in following Sec 9.8.3).

Redundancy

Redundancy N+1 redundancy is a technique for fault tolerance forseeing, in a set of N components, an extra one to
take over in case of failure.
It may be active-active or active-passive, depending on the standard status of the extra component.

Compute Redundancy

Regarding the compute, it may be protected by means of compute clustering.

Definition 9.5 (**Compute Clustering**) A technique where at least two compute systems (or nodes) work together
and are viewed as a single compute system to provide high availability and load balancing.

Enables service failover in the event of compute system failure to another system to minimize or avoid any service
outage.

Again, the implementation may be active-active or active-passive.
Also VMs Live Migration helps, since it allows to perform maintenance on a compute system without service downtime,

only some temporary performance degradation.

Network Redundancy

Link and switch aggregation are practices consisting in combining two or more physical components to make them
appear as a single logical one.
We mentioned link aggregation when speaking about LACP in Sec 4.4.4.1,

explaining that —partially— increases
bandwidth and provides a backup link in case one fails.

Aggregating two switches allows a single node to use a port-channel across two switches, and network traffic is distributed across all the links in the port-channel.

Storage Redundancy

We will go deeper in data protection in Sec 9.8.3, but these are techniques to provide straight redundancy for storage systems.

- ◇ RAID (Redundant Array of Independent Disks)
- ◇ DDS Dynamic Disk Sparing
- ◇ Erasure coding: a set of N disks is divided in M disks that store data and K disks that store coding information.

  Providing space-optimal data redundancy
- ◇ Mirrored LUNs: a LUN is mirrored to another LUN which must be in a different storage systems

Service Availability Zones

Definition 9.6 (Service Availability Zone) A location with its own set of resources and isolated from other zones

to avoid that a failure in one zone will not impact other zones

Service providers typically deploy multiple zones within a data center, and also across multiple geographically dispersed
data centers to provide high availability and fault tolerance.
There should be a mechanism that allows seamless (automated) failover of services running in one zone to another.

VM Live Migration

Recall that in a VM live migration the entire active state of a VM is moved from one hypervisor to another. We had spoken about earlier on in Sec ??, but here are anyway reported some stuff. Live migration summary:

1. copy the RAM and at the end, copy the pages writed during this phase.
2. create an empty drive on B

3. copy the CPU registers (the VM is stopped for a really short period)
4. manage vSwitch and ARP protocol. The virtual switch must be aware of the migration: if the old vSwitch

   receives a packet for the just migrated VM it should forward it to B.
5. continue running the VM on B, only when it needs the disk you stop it and start copying the disk file. A

   jumboframe can be used to avoid storage traffic fragmentation.

The whole process is a little bit easier if both the VMs use a shared storage.

Data Protection

A backup is not simply a copy of the current data to be used in case a disk fails. In case a ransomware attack occurs,
the copy will be encrypted as well, or even more trivially, if a service has a bug and it corrupts data, also the copy
would contain bad data.
“Backups must allow to travel back in time.”

The two critical terms are Recovery Time Objective (RTO) and Recovery Point Objective (RPO)3. They refer to the
time it takes to recover from a disaster and the amount of data that can be lost respectively.

Backups are typically done incrementally, meaning that starting from a full backup, then only differences are stored.
However, saving storage in this way leads to a more complex recovery process, as all the incremental backups must
be applied to the full backup to recover the data, possibly considerably increasing the RTO.
To overcome the issue, a full backup is done every now and then e.g. a week is common practice, and incremental
daily backups are done until the next full backup.

This fixes the RPO to 1 day, while the RTO still may vary depending on bandwidth, storage speed, and most
importantly size and amount changes throughout time; typically it is days (?) or hours.

Security

Information is an organization's most valuable asset.

Cloud is interesting, because, among other things, allows to distribute information in multiple places, to the cost of
possibly broadening the attack surface.
However, cloud tenants are isolated from each other, so the attack surface is typically limited to the cloud provider's
infrastructure.

The Security layer is responsible for providing secure multi-tenancy.

Three ICT Security Pillars

1. Phyisical Security
Badges, doors, locks, keys, etc...These are needed because with physical access to the hardware, one can do anything.

2. Logical Security
Accounts, passwords, firewalls, etc. . .
Logical security has been historically underestimated, but it is fundamental, and it was the easier to exploit. It refers to things like access rights, restricting an account's capabilities, etc.

3. Produceral Security
"An employee which knows the system must not be able to exploit it".

Compute

Here Physical security plays a key role, because if you have physical access to a machine, you can do anything. It

is enforced by means of badges, biometrics, disabling unused devices, surveillance. . .

Identity is a key concept in security, which in later years has changed a lot, mostly due to federated identity, which
allows to use the same credentials to access multiple services.
Associated to identity there are the long-discussed concepts of authentication and authorization.