ICT Infrastructures - Appunti

Francesco Lorenzoni

February 2024

Contents

Course info

Don't be shy to send multiple emails, prof. Cisternino receives many emails, and he known he can't reply to each one. He is okay to be contacted through teams using the symbol to "mention" him.

He designed the UniPi datacenters.

"Italy is more about the multiple micro businesses than the few existing industries"

Exam

The exam is **oral**.

Prof. Cisternino expects students to get the full picture, and understand key concepts, not to remember everything —which still wouldn't be bad ☺—.

Chapter 1

Introduction

Prof. Cisternino dropped a lot of measures in terms of Watts, Dollars, Gigabits and so on.

He mentioned with emphasis the problem of energy consumption. To give an idea, a single rack of a datacenter designed $\sim$ 10 years ago, absorbs up to $15kW$. The datacenter in *San Piero a Grado* is made up of 60 racks. It is not meant to provide the maximum energy possible for all racks simultaneously, but it still helps to get an idea of how things work in similar contexts.

1.1 Course map

1. Elements
    i. Datacenters
        (a) Power
        (b) Cooling
    ii. Cabling
    iii. Networking
    iv. Storage
    v. Compute
    vi. Virtualization
        (a) Hypervisor
        (b) Containers
2. Cloud
    i. Reference architecture
    ii. Resilience
    iii. Security
    iv. Legal aspects
        (a) GDPR
        (b) Security frameworks
    v. Procurement aspects
    vi. Operations
        i.e. Keep the system up and running while upgrading the system

Chapter 2

Datacenter

10 years ago datacenters were no more than a room with some computers, air conditioners and some plugs to power up the devices. Later on, customers started asking server vendors to include in the servers utilities to allow an *automated datacenter management*. Thus the trend moved towards **Software Defined Datacenter**, which currently is the only possible way to deploy a Datacenter. This refers to the fact that the behavior of some physical facility or system is not predefined in it's building, but to some extent it's behavior is defined by a software and so it can be changed over time, allowing for datacenters to be **future-proof**: servers may be replaced, but updating a whole datacenter is at least a 1-year project.

Datacenters **active** systems that allow to host server storage and network.

2.1 Structure

Racks are made of —~ 42[1]— *units*. Racks are typically prefabricated in groups and automatically integrated in the datacenter POD (Point Of Delivery).

Servers occupying only one rack unit are often referred to as Pizza Box

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, which must take into account dust.

However also the **floor** is not to be negliged.

- ◇ *Floating* floor or Ground floor
  - *"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 located12. This space is typically used for routing cables and for air circulation, which helps with cooling the equipment1."*[2]
- ◇ *Resistance* usually around $1\frac{ton}{m^2}$ for marble tiles, about the half for wooden ones.
- ◇ Floating floors are always a good idea.
  They provide flexibility, allow to have some sort of cable management under the racks, and in general make possible changing the layout without having to redo everything.
  When the ceiling is not sufficiently high and no floating floor is possible, the cabling is done over the servers, which is not ideal.

For example, in San Piero A Grado, there was a power cabin receiveing current from three lines. Now the whole power management components are in a container outside the building placed close to the facility.

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.

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

In theory $1V * 1A = 1W$, but in reality, performing such conversion something gets lost, so we have

$$I * V * cos\phi = W$$

[1] for 2 meters tall racks
[2] ChatGPT 4.0 - Generated

2.2 Power Management

Electric panels (aka *switchboards*) allow segmenting the power supply in the various zones of the datacenter.

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 redundacy 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.

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). ment.
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).

Definition 2.1 (Power factor) *The power factor is the ratio between the real power and the apparent power.*

Early UPSs had a 0.8 PF, but today the standard is 0.9 PF. A 100 kVA UPS would support only 0.8 kW of real power load.

Definition 2.2 (PUE) *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.

But *when* should PUE be measured?

The PUE in January is very different from the one in August, so generally it is calculated as the average of one year.

Note that a poorly designed datacenter placed in Siberia with $-20°$ may have a lower PUE than a datacenter in Italy, for instance.
In particulare geographical zones with high temperature variations over the year (e.g. in Italy the temperature variates from 40 to 50 Celsius degrees), are strongly unrecommended to build datacenters in. 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.

2.3 Cooling

Cooling is critical since its the most important factor in determining the **PUE**. Today the standard is air based[3], but there are some experiments for water based cooling.

e.g. in the Netherlands there is a datacenter which uses water from the sea to cool down the datacenter.
Prof. Cisternino displayed a Microsoft datacenter in the sea, which is cooled down by the sea itself.

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.

2.3.1 CRAC

Chillers take hot air from above and push cool air in the bottom. Then air pushed under the floating floor wants to exit, and does so going through the grates placed in front of the racks. The racks suck the cool air in front and output hot air from the back.

[3]Does not mean that water cannot be implied in the process

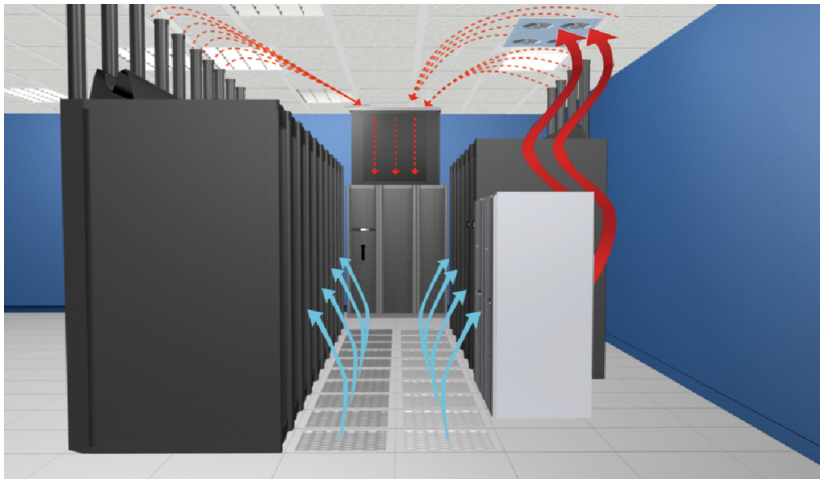

Figure 2.1: CRAC/CRAH cooling architecture

There are two **drawbacks**:

1. **Mixing air**: It is difficult to confine and keep separated hot air and cool air. The mixup between the two leads to cooling inefficiency, thus energy and money waste
2. **Absence of locality**: In case a rack has a workload heavier than others and thus requires more cooling air, the chiller must provide more cool air to all the racks in the same row; this makes this architecture particularly inefficient for datacenter which have heterogeneous workloads.

No one is using this technique today apart from Aruba, since they have very omogeneus workload. Doors are used to isolate chilled and hot air.

2.3.2 Inrow Cooling

The fan "towers" are called *inrow cooling*.

The first advantage is that it allows for heterogeneous cooling in the datacenter. Secondarily, a fan outputs hot air directly where another fan expects it to be. This allows to confine hot air and to avoid wasting energy in outputting air and sucking it.

Rack units take cool air from the front and spit it out hot from the back; inrow coolers is placed so that they suck the hot air from the back of the rack and output cool air in the front of them.

There is one inrow cooler for each *pair* rack.

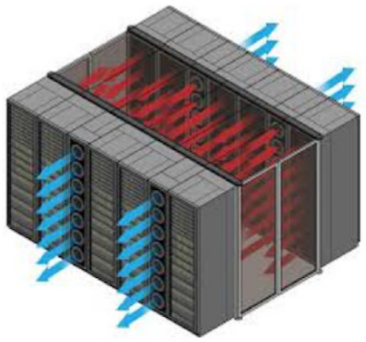

Figure 2.2: Inrow cooling architecture

2.3.3 Chilling water outside

Figure 2.2: Outside chillers

Outside of a datacenter there are chillers which cool down water which is then pumped into the datacenter, where it is used by CRAC/InRow chillers to cool down air.

It is important to ensure that the temperature does not heat up while travelling from the outside chillers to the datacenter, because it would mean wasting energy.

In SPG the outside chillers cool the water down to 18°, which seems high temperature, but in fact it is not: the datacenter is designed to work up to 26°.

The higher the "allowed" temperature is, the more is the energy saved.

Besides, **adiabatic** chillers —such as the ones in SPG— can use ***free cooling*** in case the outside temperature is lower than 18°[4], which basically exploits the lower outside temperature to *passively* chill water, without involving the compressor used in the standard cooling way.

Also **humidity** must be managed. An environment which is too dry leads to water condensation onto racks and plugs, possibly resulting in damage to devices and humans.

2.4 Redundancy for Resilience

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:

Cons

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* ☹

[4] common case in winter and autumn

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.

2.5 Cooling CPU

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 waterpowered cooling system may lead to consistent damange 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 travelling in pipes, chillers, and so on.

Most datacenters tend to have an hybrid 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.

(Woa! We need a schema...)

This is *not* the most efficient approach.

A nice question would be, *"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

2.5.1 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

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.

2.5.2 Chassis

Chassis are needed for various reasons:

- ◇ 2.4GHz is the frequency at which water in our cells resonates, and circuits generate electromagnetic fields, so it may be unsafe to directly expose humans to circuitry
- ◇ Act as Faraday cages
- ◇ TODO

Chapter 3

Fabric

"Fabric" is the term used to refer to the *interconnection* between nodes of a datacenter.
Cabling is of paramount importance.

Prof. Cisternino learnt it "the hard way" when he performed the cabling of the first UniPi datacenter by himself

1. Maintenance
2. Cooling
    i. Cables may heat up
    ii. Cables may obstruct air flow
3. Determines which machines interact with each other (*fabric*)
4. Bandwidth
5. Not neglectable cost

We refer to North-South traffic indicating the traffic outgoing and incoming to the datacenter (internet), while we refer to East-West as the internal traffic between servers. Most of the network (or fabric) traffic is processed horizontally (North-South traffic)[1].

3.1 Bandwidth and Storage implications

A standard datacenter has servers connected with 25Gbit links in both directions, summing up to 50Gbit total bandwidth. Current SSDs using NVMe provide much more, about $3.5GB/s$, making 4 drives are enough to saturate a 100Gbit/s link.
We moved from a situation where the **bottleneck** were slow Hard Drives, to the current one where the bottleneck is the —network— **bandwidth**.
Recently the PCI 3.0, which lasted very long —providing roughly$\sim$ `120Gbit/s` using 16 pin—, suddenly become unsufficient to handle the needed traffic.

Considering this, datacenters must be designed to allow *Terabytes* of data to be moved in east-west traffic.

*The **fabric** is the glue that makes the datacenter possible.*

Besides, a single server is *unable* to handle 10TBs of data and handling requests from 3000 users simultaneously. It is necessary to **distribute** the requests.

HDDs are still currently used for **cold storage**; CPUs will access data exclusively from SSDs, and sometimes the server is shipped with on board **full-flash storage**.
The difference in price between SSDs and HDDs becomes negligible since you pay for top CPU, top GPU, top RAM; furthermore, you can't waste —the high amount of— energy —consumed by such components— by waiting for a slow drive.

SSDs have a known write limit, but today, the usually last enough time: if you write the whole disk every day it will last for 5 years. Most-likely after five years you'd have to renew some components anyway, besides the failure is a predictable event.

[1] Seems odd that "horizontal" refers to North-South traffic, but that's how it is.

3.2 Cables and standards

3.2.1 Optical

Electric current propagates at a speed $s = \sim 0.6c$. Hence **optical fiber** is —at least in theory?— faster.

Lasers are a coherent beam of equal fotons. It is possible to transfer energy through such fotons. Something resembling a laser is used for optical fibers.

Blu-Ray came out when scientists managed to create light using frequencies in the Blu area, which are the higher ones. Currently, the best and most expensive optical fibers exploit blu-lasers as source of light.

Note that with optical you always need 2 fibers, one sending and the other receiving. The two possible connectors are `SC` and `LC`. Sometimes the two ends of the cable are detachable so that the cables may be switched; this is useful because sometimes you may want to attach the TX cable on the RX plug and viceversa.

Figure 3.1: `SC` and `LC` connectors

3.2.2 Copper wires

In case of electricity there are many aspects to be considered. Interferences, cable diameter/size, length, and also the fact that if a `1` has been transmitted for some time, it takes longer to transmit a `0`, due to the *commutation* that must happen.

RJ45 is a standard physical interfaced for copper wires, which allows up to 1Gbit regularly. The `Cat 7` cables still use the RJ45 as connector and provide instead 10Gbit/s, but are very uncomfortable, they are so thick that they are difficult to bend.

It is estimated that there have been installed $70 \times 10^9 m$ of Ethernet cables, making them the most used.

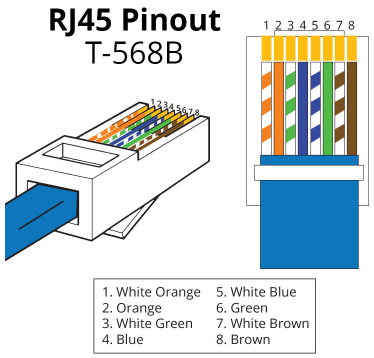

Figure 3.3: RJ45 - T568B

3.2.3 SFP - Small Form-factor Pluggable

There can be a cable with a LC in one side and a SC on the other side. Instead of making switches with the optical plugs, switches were created with electrical plugs that would be able to host a **standard transceiver**. The latter is a pluggable module that will receive current power and electrical signals for the transmission, which is responsible for transitioning between electrical signals and Optical signal (and viceversa).

The aim of SFP is to decouple the optical transceivers from the server modules.

Is this correct?

They allow to go *optic-copper*, *copper-optic*, *optic-optic* and *copper-copper*.
SFP and GBIC (oldest one, now dead) pluggable modules acting as active transceivers for optical wiring using RJ45

connector.
A single cable having SFP ends costs about 100€. The cost ain't neglectable ☺.

`SFP` $\longrightarrow 1Gbit$
`SFP+` $\longrightarrow 10Gbit$
`SFP28` $\longrightarrow 25Gbit$
This is the current standard
`QSFP28` $\longrightarrow 4 \times 25Gbit$

Figure 3.4: SFP transceivers form factors

Fun fact: ci sono 9 cavi USB-C e solo due portano informazioni video.

Issues about cabling and fabric

The key point is that it would be desirable for cabling to be reconfigurable, that's why transceivers are so important.

There are things called *"Muffole"*, which are used for joining optical fiber cables, allowing for longer distances to be covered. They are designed to be underground.

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` respecitively, and are used for north-south traffic.

The `Q` letter stands for *Quality*

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.

In every datacenter it is *MANDATORY* to document the cabling.

3.2.4 InfiniBand

Even though Ethernet is famous, it is not the only standard. InfiniBand is another one, which is used in supercomputers and known for its very high throughput and very low latency ($\sim 2\mu s$). It may send messages up to 2GB each, with 16 priority levels. It is a *lossless protocol*, meaning that if a packet is received, its integrity is guaranteed.

IB avoids TCP/IP stack, which is very heavy, and instead uses MPI (*Message Passing Interface*), which is a way to distributed parallel programs, also exploited by OmniPath.

InfiniBand cables and connectors look similar to Ethernet ones, but they are not compatible.

Aside from HPC environments, it is uncommon to build an entire network with InfiniBand, typically there is an IB switch to whom the servers equipped with IB NICs are connected, which intereacts with the rest of the network with Ethernet, because "it's cheaper and it works".

Today, we are about 400 Gbit/s on both IB and Ethernet.

3.2.5 RDMA - Remote Direct Memory Access

RDMA is a technology API based (not a protocol!) that allows to access memory of a remote machine without involving the CPU or the OS of the remote machine.

RDMA supports zero-copy networking by enabling the network adapter to transfer data directly to or from application memory, eliminating the need to copy data between application memory and the data buffers in the operating system. The main use case is distributed storage.

RoCE (*RDMA over Converged Ethernet*) is a network protocol that allows remote direct memory access (RDMA) over an Ethernet network.

3.2.6 Omni-Path

Omni-Path is a high-performance computing network architecture, developed by Intel. It is a successor to Intel's InfiniBand, and competes with InfiniBand's EDR and HDR technologies.

Intel plans to develop technology that will serve as the on-ramp to *exascale computing*[2], which is the next frontier in high-performance computing.

[2]A computing system capable of the least one exaFLOPS

Chapter 4

Networking

The two key aspects of a network are:

1. **Bandwidth** $\longrightarrow$ amount of data per second that can be moved through a specific connection
2. **Latency** $\longrightarrow$ 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.

Latency —in a datacenter— to transmit data on the cable using *"pure ethernet"* is of the order of $0.5 \times 10^{-6} s(\mu s)$. If the TCP/IP stack is used (standard application case), latency is about $70 - 90 \mu s$.

Furthermore, current drives have reached speeds such that latency may act as bottleneck between them and the CPU.

Cable aggregation (e.g. aggregating 4 cables 10Gbit/s, providing 40Gbit/s total)can be performed only at a low —physical— level. Otherwise the TCP/IP stream will be associated to a single cable of the ones aggregated, resulting in less bandwidth.

Latency-sensitive

Some workloads are called *latency-sensitive*, making the latency introduced by TCP-IP stack a problem.
Inside a datacenter nowadays the typical latency is sub microsecond.

Regarding this issue, technologies mentioned earlier come in handy; *InfiniBand* is a fabric technology that allows to have a very low latency, and is used in HPC[a] environments, and *OmniPath* is a technology that is similar to InfiniBand, but is more scalable and is its natural successor. Also *RDMA* and *RoCE* are technologies that allow to access memory of a remote machine without involving the CPU or the OS of the remote machine, bypassing the TCP/IP stack.

Fibre Channel switches are used in storage area networks, and are used to connect storage to servers CPUs. They are used in datacenters, but not for networking.

[a] High performance computing

4.1 SDN - Software Defined Networking

SDN is a new approach to networking that uses software-based controllers or application programming interfaces (APIs) to communicate with the underlying hardware infrastructure and direct traffic on the network.

In general a Software Defined Approach aims to abstract all the infrastructure components (compute, storage and network), and pools them into aggregated capacity.

When such approach is applied to a whole datacenter, it is called **Software Defined Datacenter**, and it is a way to abstract all the infrastructure components in order to provide IT as a service.

The problem was that the network infrastructure was "ossified" and not programmable. SDN allows to program the

network, and to make it more flexible and adaptable to the needs of the applications, without having to disrupt the existing infrastructure.

The key idea proposed in the OpenFlow article, which eventually became a standard, is to separate the control plane from the data plane, and to have a controller that can program through an API the data plane, where the **Flow Table** resides.

An interesting use of OpenFlow was implemented by a University and called Sandwich firewall, which consisted in routing the first part of the stream through a firewall and if the stream was not malicious, it was routed directly to the destination, otherwise it was dropped.

4.1.1 Hyperconverged Infrastructure (HCI)

HCI is a software-defined IT infrastructure that virtualizes all of the elements of conventional hardware-defined systems. HCI includes, at a minimum, virtualized computing (a hypervisor), a virtualized SAN (software-defined storage) and virtualized networking (software-defined networking).

4.2 Layers

Programmers usually do not care about anything under layer 3/4 traffic. However, in datacenters it is fundamental to understand how layer 2 works.

Also because in datacenters there are no routers doing the work for you; you are building the fabric in the first place.

Layer 2 is fundamental for 2 reasons:

1. East-west is Ethernet in the datacenter
2. All the dozens of protocols used in swithces are really used, so they are important.
3. MTU - Maximum Transmission Unit

4.2.1 Protocols inside switches

- LLDP Link Layer Discovery Protocol - Allows to reconstruct at least partially the functioning of the network.
- DCBX Data Center Bridging Exchange - A meta-protocol so that two devices can agree on the configuration of a bunch of protocols, tipically related to storage/data
  e.g. "I need 50% percent of the bandwidth otherwise I can't work".
  It represents part of some kind of QoS for Ethernet
- PFC Priority Flow Control
- ETS Enhanced Transmission Selection
- RSTP Rapid Spanning Tree Protocol - Uses BPDU packets to explore the graph of the network and compute the spanning tree of the network and detect the —malicious— cycles if any.

This just to recall that the switch is not a stupid thing! It is complex, fascinating, and deserves love; it's crucial to understand its functioning, also because its protocols occupy bandwidth.

4.3 Ethernet Topology

Typically nowdays the network is a **graph**, where internal nodes are switches or routers, and the leaves are servers.

The phyisical medium is no more shared, but conceptually the data link layer behaves as if it was.

On a switch, the only way to emulate a **shared bus**, is to ***"copy-paste"*** a frame onto multiple ports, losing the "identity" of frames; there is no routing table at layer 2. Packets in higher layers (IP?) have an ID, but frames don't, making it impossible to recognize whether a frame is a copy of another one or not. This approach makes **loops** a problem, because they disrupt performance by generating a packet storm.
The solution would be to ensure that the topology resembles a **tree**, instead of a graph. But, at the same time, a **fully connected graph** allows to have multiple routes for the same destination, possibly enhancing performance, reducing "hops" before reaching the destination.

4.3.1 `RSTP`

So... how can we leave the graph to be connected, but making it a tree from a logical point of view?

The answer is the `RSTP` protocol.

RSTP sends *probes* to understand whether there are loops and where are PCs located. In case of link failure, RSTP is able to adjust the logical tree, blocking the link that caused the loop.

RSTP can be used in campus networks or other networks exhibiting primarily North-South traffic, but it is not suitable for datacenters, where the traffic is mainly East-West: the protocol is too slow (order of *seconds*)to handle the high number of links and the high number of switches typical of a datacenter.

4.3.2 Network Chassis architecture

Historically a typical solution was to use a **network chassis**, consisting of many *line cards* (blade servers?), each being a switch. All links from various racks converged in the chassis, and the chassis was responsible to route the traffic to the right destination, ultimately resembling a **star topology**.

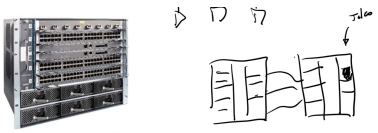

Figure 4.1: Network chassis

The key advantages were:

- ◇ Possibility to buy a switch with few line cards and then add more as needed, reducing inital CAPEX.
- ◇ Chassis act as a single switch, so they are much easier to manage than a bunch of switches.
- ◇ Chassis includes redundancy mechanism to allow *hot-replacing* of a line card.

The Chassis is costful since it is a complex modular structure, so people tried to manage a stack of switches using protocols, making them act as if they a chassis. The chassis provided redundancy for every component, and included —two ☺— RPMs, routing processing modules.

Ports were numbered using $\langle line, port \rangle$. The line number was used to identify the line card, and the port number was used to identify the port on the line card. According to Cisternino this is important, and there is a "lot of legacy".

Chassis died when moving from 10Gbit/s to 25 and upper, because it was impossible to design a chassis the could allow hot replacing and modularity, and at the same time allow high speed links. The size of the lanes and the type of the connector would limit the bandwidth and the frequency the physical link can resist.

Nowdays star solutions are rare to find. Starting from 2013/2014 it was not convenient to use chassis anymore, because they would get old too soon. Swtiches started to have a fixed standard *non-modular* form factor.
All vendors embedded proprietary protocols inside switches to allow to manage two switches as if they were one, and to aggregate two ports (one port for each switch) to have a single logical link, called **port channel**. The standard protocol to do so, is called LACP (Link Aggregation Control Protocol) and is described in 4.3.4.

The port channel is a single logical link, but the bandwidth is *not aggregated for the same data stream.* However, the logical link provides redundancy and avoids loops.

4.3.3 Three-tier architecture

Simple architecture consisting of

1. Core switches
2. Aggregation switches
3. Access switches

The switches are connected in a tree-like topology, with the core switches at the top, the aggregation switches in the middle, and the access switches at the bottom.
STP is used to prevent loops in the network.
Also provides active-passive redundancy which leads to inefficient east west traffic, because the traffic is forced to go through the core switches (?), and devices connected to the same port may contend for bandwidth.
Moreover communication server-to-server might requires crossing between layer, causing latency and the above-

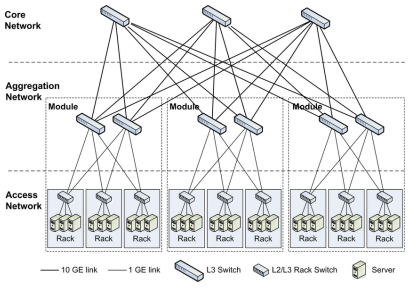

Figure 4.2: Three-tier architecture schema

mentioned traffic bottleneck.

This architecture is not used anymore, because it is not scalable, and it is not able to handle the high number of links and switches typical of a datacenter; more specifically, it does not fit virtualization most crucial need to be able to freely move VMs between servers.

4.3.4 Spine Leaf architecture

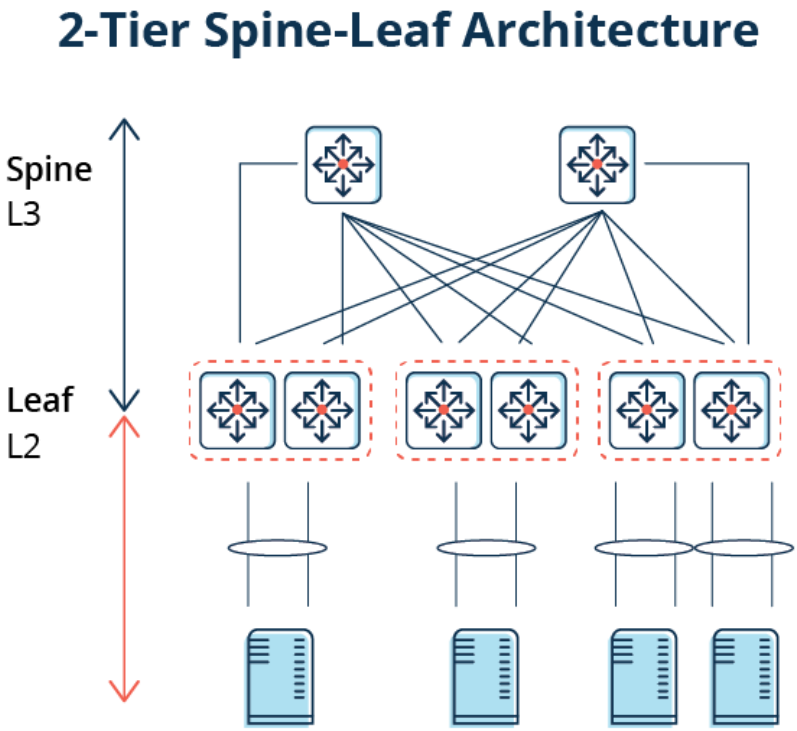

Figure 4.1: Spine-leaf architecture schema (from Arubanetworks.com)

A **spine-leaf** architecture is data center network topology that consists of two switching layers:

1. **Spine layer**
   Swtiches responsible for routing traffic, working as the backbone of the network.
2. **Leaf layer**
   Switches connected to endpoints, such as servers, storage devices, firewalls, load balancers, edge routers, etc.

Since every leaf switch is connected to every spine switch, the spine-leaf architecture is a **fully connected** network, ensuring that any source is always the same number of hops (actually only two or four ☺) away from any destination, so latency is lower and predictable (fixed).

"However, it's not true that every rack is connected with any other rack. There is some topology that has been designed with the data center and it will." -Cisternino

Capacity also improves because STP is no longer required. While STP enables redundant paths between two switches, only one can be active at any time. As a result, paths often become oversubscribed. Conversely, spine-leaf architectures rely on protocols such as *Equal-Cost Multipath* (`ECMP`) routing to load balance traffic across all available paths while still preventing network loops.

Spine-leaf allows *scale-out* opposed to *scale-up*, by adding additional spine switches, ultimately increasing capacity in case the bandwidth is not enough; doing so reduces also the subscription

LACP

Loops are prevented using `LACP` (*Link Aggregation Control Protocol*), which is a protocol that allows to aggregate multiple links into a single logical link, providing higher bandwidth and active-active redundancy (in case a link fails); it also ensures no loops because each link is a single channel, and these are named *port channels*.

LACP also provides a method to control the bundling of several physical ports together to form a single logical channel.

Note that even though the bandwidth is aggregated (i.e. $2 \times 25 Gbps$), the single stream is still limited to the bandwidth of a single link (i.e. $25 Gbps$), because the traffic goes only from one way to the other each time.

Advantages of Spine-Leaf

- ◇ Modular: you can mix and match devices with fixed size switches.
- ◇ Latency predictable: every host is distance one or two hops to each other host.
  Actually it is 2 or 4 hops, because we have either
  - server → leaf, leaf → server
  - server → leaf, leaf → spine, spine → leaf, leaf → server

  However, the hops from server to leaf and viceversa are typically not counted as "hops".
- ◇ Bandwidth control: it's possible to chose the proportion of NS and EW traffic and overbooking.
- ◇ Active-active redundancy: both links of the port channels are enabled, so is the LACP to decide.
- ◇ Loop aware topology: a tree topology with no links disabled for redundancy reasons.
- ◇ Interconnect using standard cables (decide how many links use to interconnect spines with leaves and how many others link to racks).

With this architecture it's possible to turn off one —spine?— switch, upgrade it and reboot it without compromising the network. Half of the bandwidth is lost in the process, but the twin[1] switch keeps the connection alive.

Just a small remark: with spine and leaf we introduce **more hops**, so more latency, than the chassis approach, which basically represents a star topology, with the big network chassis as the center of the star. The solution for this problem is using as a base of the spine a huge switch (256 ports) which actually acts as a chassis, in order to reduce the number of hops and latency.

Oversubscription

Oversubscription is the practice of connecting multiple devices to the same switch port to optimize the use. For example, it is particulary useful to connect multiple slower devices to a single port to take advantage of the unused capacity of the port and improve its utilization. However, devices and applications that require high bandwidth should generally connect with a switch port 1-on-1, because multiple devices connected to the same switch port may contend for that port's bandwidth, resulting in poor response time. Hence, significant increases in the use of multi-core CPUs, server virtualization, flash storage, Big Data and cloud computing have driven the requirement for modern networks to have lower oversubscription. For this reason, it is important to keep in mind the oversubscription ratio (downlink ports —to servers/storage— to uplink ports —to spina switches—), when designing the fabric. Current modern network designs have oversubscription ratios of 3:1 or less, meaning that bandwidth for NS traffic in $\frac{1}{3}$ of the bandwidth for EW traffic.

[1] What does twin mean? Another spine?

Uplink and Downlink ratio

"Uplink" and "downlink" refers to the direction of the link in the topology. It appears more clear with a picture.

The ratio between uplink and downlink ports is important because it determines the bandwidth available to the servers. Typically the desired ratio is 1/3 of bandwidth uplink and 2/3 downlink. So, considering the following switch having 20 QSFP ports (`40Gbit/s`) and 4 QSFP28 ports (`100Gbit/s`), we have:

- 800 Gbit/s downlink, exploiting 20 QSFP
- 400 Gbit/s uplink, exploiting 4 QSFP28

Even though traffic is categorized as NS or EW, there is also the *aggregation traffic*, due to MLAG

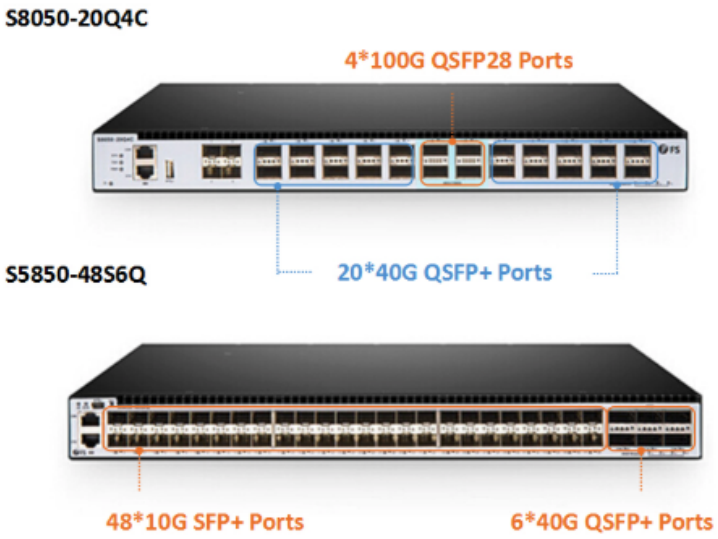

Figure 4.2: Blue ports are downlink, yellow ports are uplink

Considerations on Increasing Bandwidth

In case the uplink bandwidth is not enough, it is possible to increase it by:

- adding more spine switches, and connecting them to the leaf switches.
  - More redundancy in the paths between leaves, allowing to have even paths to the same destination
  - More uplink ports needed, reducing oversubscription ratio
- adding leaf switches, and redistributing the servers among them.
  - Less downlink ports used per switch, reducing oversubscription ratio
  - *¡ACHTUNG!* Spine switches may need to be upgraded to support more leaves
- adding a *super-spine*, enhancing the number of hops but allowing to handle a huge number of servers.

The *super-spine* fits an "extreme" scenario: the number of ports in spines is insufficient to handle the number of leaves.
Let's make a few calculations: Considering that LACP allows to aggregate a maximum of 8 ports, but it is very unlikely to aggregate 8 ports from spines to leaves, since it would mean that every leaf switch has 8 ports for each spine.
Assuming to have respectively as spine and leaves the switches in Fig 4.2 and the oversubscription ratio to be 1:2, the maximum number of spines would be 6, and the maximum number of leaves 20. Each leaf has 48 downlink ports, which however may be aggregated in 12 groups of 4 ports, hence 12 servers per leaf.
Summing up:

- 6 spines
- 20 leaves per spine
- 12 servers (at least) per leaf
- $20 \times 12 = \mathbf{240}$ servers units.

UniPi datacenter has 62 physical nodes, just for comparison. Even assuming to halve the number of spines and to aggregate every pair of links from spines to leaves, the number of servers would be 240, still a pretty big number.

How can a leaf know which is the best path to send its data onto?

We want the path which is, at the moment of transmission, less used by other leaves and with more available bandwidth.

Answered by Copilot ☺

In a spine-and-leaf architecture, the decision on the best path for data transmission is typically made using routing protocols such as Equal-Cost Multipath (ECMP).

ECMP is a routing strategy that allows traffic to be distributed across multiple paths of equal cost. This helps to balance the load across the network and utilize all available bandwidth. When a leaf switch has data to send, it uses ECMP to determine the best path among the multiple paths available.

The decision is based on various factors such as the number of hops, path cost, and current network congestion. ECMP can dynamically adjust the traffic distribution based on the network conditions, ensuring that the data is always sent on the optimal path.

So, in a spine-and-leaf architecture, the leaf switches don't need to know the specific usage of other leaves. They rely on ECMP to determine the best path for data transmission. This allows the architecture to efficiently handle traffic and provide high performance, even in the presence of heavy east-west traffic.

4.3.5 Full fat tree

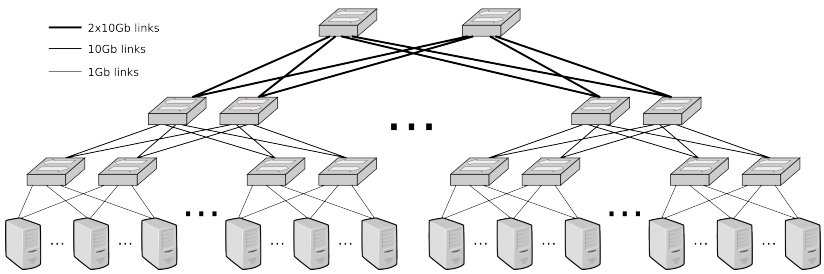

Figure 4.3: Full fat tree network topology schema

In a **full fat tree** topology, branches nearer the top of the hierarchy are "fatter" (thicker) than branches further down the hierarchy. In a telecommunications network, the branches are data links; the varied thickness (bandwidth) of the data links allows for more efficient and technology-specific use, a typical use case is for HPC.

Full-fat tree is rarely needed.

The full fat tree resolves the problem of over-subscription. Adopting the spine and leaf there is the risk that the links closer to the spines can't sustain the traffic coming from all the links going from the servers to the leaves. The full fat tree is a way to build a tree so that the capacity is never less than the incoming traffic. Since it's quite expensive some oversubscription can be accepted.

Definition 4.1 (Full Fat Tree) *"The full fat tree is simply a spine and leaf architecture in which the bandwidth that you reserve to East West traffic is equivalent to the bandwidth that you reserve between the North and South uh traffic."*

Hence, the oversubscription ratio is 1:1.

4.4 Virtualization

What's the key difference between Layer 2 and 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.
But luckily, **VLANs** exist.

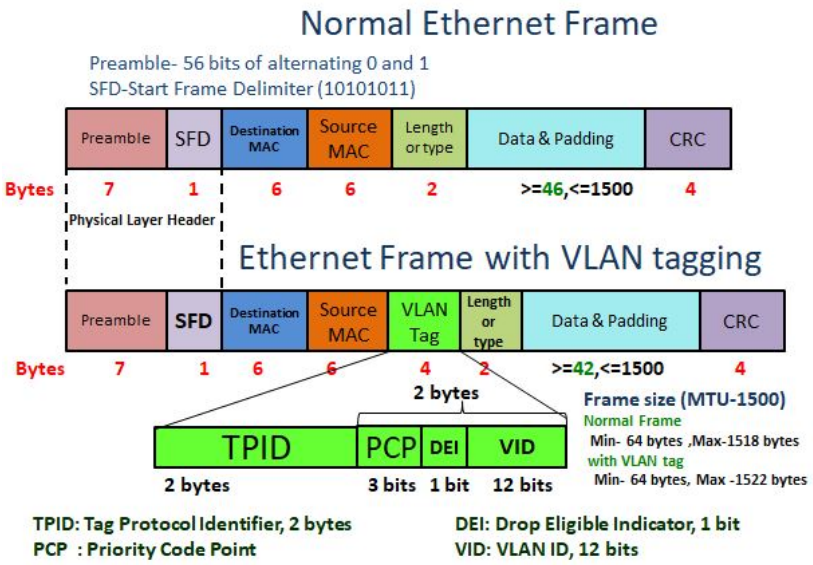

Figure 4.4: Ethernet frame with VLAN header

With VLAN frames are extended by 4 bytes[2]. 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.

A port can be in three states (CISCO Terminology):

- **Access Mode**: An access port can have only one VLAN configured on the interface; it can carry traffic for only one VLAN1. It connects an end device to the network and works only in a single VLAN3.
- **Trunk Mode**: A trunk port can have two or more VLANs configured on the interface; it can carry traffic for several VLANs simultaneously1. Trunk mode is designed for connecting devices using tagged VLANs (e.g., VLAN-enabled switches and NICs). Ports using trunk mode can be linked between various network devices and are capable of carrying traffic across multiple VLANs4.
- **General Mode**: General mode allows multiple untagged VLANs and also multiple tagged VLANs to exist on the same switch interface2. While it is possible to have multiple untagged VLANs on a General port, you can only have ONE (1) PVID (Primary VLAN Identifier). The PVID represents the native VLAN. Untagged traffic may be sent via several untagged VLANs, returning untagged traffic will only be received by the PVID and therefore will NOT be forwarded to a specific VLAN2.

4.5 Network Administrator POV

The switch is split in two planes:

- **Control Plane**
  This plane is necessary to configure the data plane to make it behave according to our needs. Here there is an *OS*, which used to be proprietary with a functioning fitting a specific network configuration, but nowdays they

[2] 12 bits are reserved for the VLAN ID allowing up to 4094 ($4096 - 2$) logical partitions

are usually more configurable and may even be *open* OS.
Dell's switches now have an *open* OS on board.

- ◇ **Data Plane**
  Here lies the chip responsible to perform all the data link operations required, runs protocols, handles VLANs, etc.
  OpenFlow allows us to manage the flow table inside of a switch.

The two planes are linked by a low-bandwidth PCIe.
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.
Prof. Cisternino performed a demo of this in class.
Interestingly, the behaviour of the `netsh` command in Windows is very similar to the one of a switch.

Chapter 5

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****."* -Prof. Cisternino

Data is the most important of a system. Since data loss is **permanent**, the storage is completely different from computing or networking.

Historically the storage was the slowest part of the system, *ms* against *ns* of the CPU. Today, with SSDs, the gap is considerably reduced to μs, they are $\sim 100x$ times faster. NVMe stands for *Non-Volatile Memory Express*, and is a protocol (*not a HW component!*) that allows to access the storage directly from the PCIe bus, without having to go through the SATA controller. This allows to have a much higher throughput, and a much lower latency.
Optane was a technology developed by intel which is now end of life

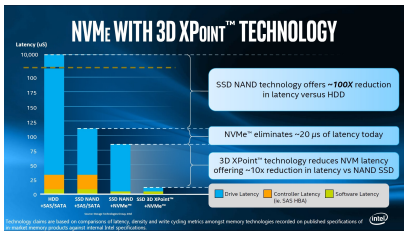

Figure 5.3: Storage types comparison
NVMe basically removes the orange part of the figure, which is the latency introduced by the controller, since it is a *controller-less protocol* and allows to access the storage directly from the PCIe bus.

Why would a 15TB disk be better than a 27TB disk?

Assume the same performance, and the same price.

It would be preferrable because it would take less time to extract all the data from the disk[1], since it is smaller.

However, large capacity drives are used for *cold storage*, where the data is not accessed frequently, speed is not a priority, and even if the data is accessed, only a portion of the disk is needed at a time; in case of failure and thus needing to retrieve an entire backup, the time taken to retrieve the data is not a priority, since this —hopefully— happens only "once".

5.1 SSDs - QLC and TLC

SSDs were invented by Toshiba back in 1980, but they were not popular for almost 30 years, until they eventually became cost-effective. Sometimes extra size in SSDs is used for redundancy, to increase the lifespan of the disk e.g. on a 30TB disk, only 10TB are used, the rest is used for redundancy, extending x3 the lifespan of the disk.
DWPD stands for *Drive Writes Per Day*, and is a measure of how many times the disk can be written to in a day. It can be calculated as $\frac{TBW}{365\times Years\ of\ Warranty\times capacity}$
.

[1] i.e. taking advantage of the space provided

TLC stands for *Triple Level Cell*, and QLC stands for *Quad Level Cell*. The difference between the two is the number of bits stored in each cell. The more bits stored in each cell, the cheaper the disk is, but the slower it is. The more bits stored in each cell, the more difficult it is to read and write the data, and the more difficult it is to keep the data stored in the cell.
Generally QLC disks are used for cold storage, while TLC disks are used for hot storage. TLC in general is more reliable than QLC, has a longer lifespan and better performance, however they cost more.

5.2 Storage Concepts

5.2.1 Tiering - Memory Hierarchy

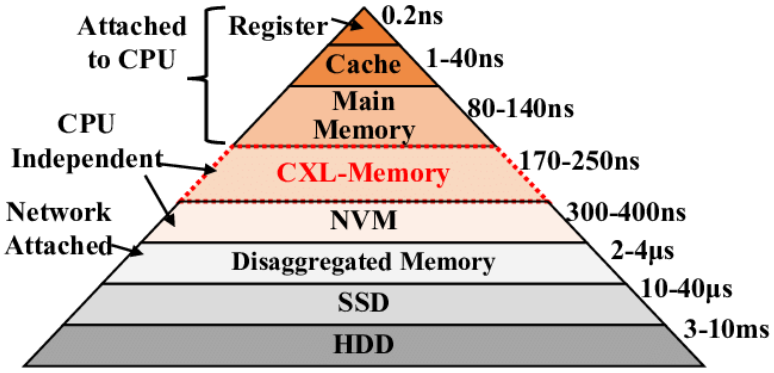

Figure 5.1: Memory tiering hierarchy
Ram could actually be split in `RAM` and `nvRAM` (Non-volatile `RAM`, uses `nvDIMM`), which is used to store the data in case of a power failure. Sometimes, *tape* is included in the hierarchy, because it is used for long-term storage, and it is very cheap.

Tiering consists in categorizing the data in different categories, and storing the data in different types of storage, depending on the category. The data that is accessed more frequently is stored in the fastest storage, while the data that is accessed less frequently is stored in the slowest storage. This allows to increase the performance, and to reduce the cost.

5.2.2 IO operations, are they all the same?

IOPS (Input/output operations per second) is an input/output performance metric used to characterize computer storage devices; it is associated with an access pattern: *random* or *sequential*.

Random vs Sequential access

Before explaining the distinction, is important to remember the concept of *queues*: for each thread, the OS can implement a series of queues to solve asynchronously the I/O requests. Using multiple queue can make performances better, since having the OS to manage parallel requests will increase throughput. If the queries are latency sensible, not using a queue is better, since usually queue requests overlap between each other.
Random access files are advantageous in scenarios where frequent direct access or modification of specific records is required, while *sequential access files* are advantageous in scenarios where frequent reads of the full files are required. The disk behaves differntly in case of access of those files.
To have a full picture of random vs sequential access, check this site: https://www.prepbytes.com/blog/general/difference-between-sequential-and-random-access-file/

Cisternino's demo

Prof. Cisternino showed a demo in class, where he used a tool to measure the IOPS of a disk.

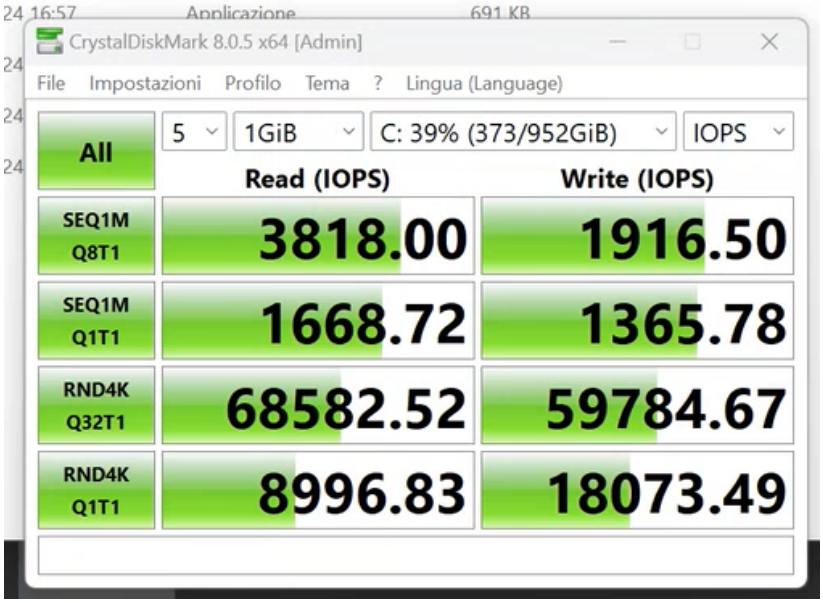

Figure 5.2: IOPS demo; respetively

Recall that IOPS by itself is meaningless! It is a number that must qualified and put in relationship with something else.

5.2.3 Latency and Storage Aggregation

A mechanical hard drive introduces 2.71% of latency when reading, for instance, 40MB of data. Optane can perform 416 accesses in the same time needed by a mechanical hard drive to perform 1 access. It looks like the latency in this latter case is neglegtible. Someone may be tempted to reduce the size of read/write operations and perform multiple smaller ones, since "it's free".
Latency in general is due to:

- **Software**
  μs order which cannot be removed
- **Controller**
  Taken down to $20\mu s$ with NVMe (even $2.8\mu s$ according to Copilot)
- **HDD latency**
  This was drastically reduced with SSDs and got even less with 3D NAND.

Latency may be solved by **storage aggregation**, which consists in aggregating multiple storage devices into a single logical unit, in order to increase the performance and reliability. Even if the data is split in multiple disks, the whole system is "pictured" as a single huge drive[1], making a huge difference in terms of latency, since multiple `read/write` requests may be sent in parallel to multiple disks.

[1] *"Cloud resource pooling"* rings a bell?

5.2.4 Storage Fabric - Fibre Channel

Fibre Channel is the fabric dedicated to storage; the link coming from the storage ends up in the *HBA* (Host Bus Adapter) in the server.
The idea is to have an interface which announces itself as drive and that manages the remote storage through Fibre Channel. Fibre Channel typically runs on optical fiber cables, but may also run on Ethernet cables (FCoE).
In Fig. 5.3 is depicted the ideal architecture for Fibre Channel, where the storage is connected to the network through a switch, and the servers have a dedicated HBA to connect to the storage.

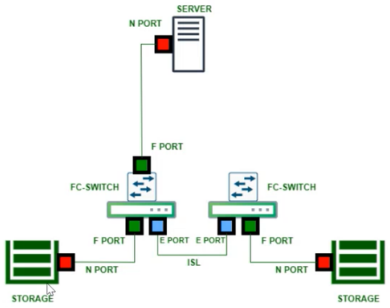

Figure 5.3: Fibre Channel desired architecture

5.2.5 Bus, controller and some numbers

A bus is a component to whom multiple devices may be attached. It has a clock and some lanes, 16 in the case of PCI, each one providing almost 1GB bandwidth, summing up to $\sim 15GB$: 4 drives are enough to saturate a full PCI bus, or a 100Gbit link ($12.5GB/s$).; in fact an NVMe SSD has a bandwidth of 3.5GB/s, hence $3.5 \times 4 = 14GB/s \simeq 15GB/s$.
NVMe is often used in the lower memory tier of the RAM: its speed is only one order of magnitude less than RAM, but can provide high capacity without any problem. It may represent a valid super-fast cache level for the RAM and hence started being associated in one single level to implement a big RAM tier, in a totally transparent way for the system.
Since the software latency in disk IOs is 5 microseconds more or less, TCP/IP software introduces also a latency of 70-80 microseconds, the disk is no more a problem. Indeed, the problem is now the network, not only for the latency, but also for the bandwidth: as stated before 4 NVMe totally saturate a 100 Gbps link.

5.3 Redundancy and backup

5.3.1 Checkpoints

It's unpractical for a system to go down after 5 months. For this reason it is necessary to have checkpoints, which are points in time where the system can be restored to. The system can be restored to the last checkpoint, and the data that was written after the checkpoint can be re-applied. This is similar to what happens to applications on smartphones are closed and then re-opened, the application is restored to the last checkpoint.

5.3.2 RAID

RAID stands for *Redundant Array of Independent Disks*. It is a technology that allows to combine multiple disks into a single logical unit, in order to increase the performance, the reliability, or both. There are different levels of RAID, each with different characteristics.
Historically *Redundant Array of Inexpensive Disks*, because it was more common for disks to eventually fail, so RAID was the only countermeasure to this. Today, disks are more reliable, so RAID is used more for performance reasons.
In RAID, **XOR** is used to calculate the parity of the data. The parity is used to recover the data in case of a disk failure. The parity is calculated by XORing the data of the disks. The parity is stored on a separate disk, called the parity disk. The parity disk is used to recover the data in case of a disk failure.

5.4 Network sharing architectures

Before going into the details of the architectures, it is important to understand the difference between **protocols** and **architectures**.
SMB/CIFS is a protocol that allows to share files over the network. It is used by Windows, but it is also supported by Linux and MacOS. NFS is a protocol that allows to share files over the network, it is used by Linux and MacOS, but it is also supported by Windows.
NFS is faster than SMB, but it is also less secure. SMB is slower than NFS, but it is also more secure.
These however are protocols for file sharing, not properly "architectures".

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

5.4.1 File based - NAS

NAS are devices that are connected to the network, and that are used to store files, providing aggregated capacity. They are used to store files that are accessed by multiple users, and that need to be accessed from multiple devices. NAS systems have integrated HW and SW component, including CPU, memory, NICs, optimized OS for file serving, file sharing protocols and so on.
Typically they exploit SMB/CIFS or NFS protocols, or AFP over optical fiber, and represent a good solution for *document management*.

5.4.2 Block based - SAN

SAN stands for *Storage Area Network*, which enables the creation and assignment (i.e. access and share) of storage volumes to compute systems.
The compute OS (or hypervisor) discovers these storage volumes as local drives. The servers have different NICs (HBA) connected (usually through fiber channels) to those blocks, which are aggregated volumes.
SAN also enables performance optimization of the storage by performing deduplication (delete sequence of blocks that are equivalent).
HBA stands for *Host Bus Adapter*, and is a device that allows to connect a computer to a storage device. It is used to connect a computer to a storage device, and to allow the computer to access the storage device.
SAN was, before SSDs, one of the datacenter pillars. Its architecture included a "Head" to which drives were attached, and the head was connected to the network. The head was used to manage the drives, and to allow the servers to access the drives.
When SSDs became popular, the head became a bottleneck, because it was not able to keep up with the speed of the SSDs(Recall that 4 SSDs are enough to saturate a 10Gbit link, See Sec. 3.1). For this reason, the head was removed, and the drives were connected directly to the network. This is called **DAS**.
However with groups of mechanical drives, —if the data is splitted in a smart way— it's possible to be faster of a single SSD, since the request will be forwarded in parallel to different drives.

Protocols

SANs are classified based on protocols they support. Common SAN deployments types are Fibre Channel SAN (FC SAN), Internet Protocol SAN (IP SAN), and Fibre Channel over Ethernet SAN (FCoE SAN), ATA over Ethernet (AoE) and HyperSCSI (). It can be implemented as some controllers attached to some JBoDS (Just a Bunch of Disks)
While NAS provides both storage and a file system, SAN provides only block-based storage and leaves file system concerns on the "client" side. However, note that a NAS *can* be part of a SAN network.

Pools and LUNs

Storage pools are used to combine multiple storage devices into a single logical unit, in order to increase performance and reliability.
The SAN is divided in different Logical Unit Numbers (**LUN**s), which abstract identity and internal functions of storage devices, and appear as phyisical storage to the compute system.
Storage LUNs define a storage partition and are used to assign storage —a portion of the pool— to a server, and to allow the server to access the storage, using ACLs.

In the following section, some LUNs features are listed

- Storage **capacity** of a LUN can be dynamically expanded or reduced by means **virtual storage provisioning**, i.e. present a LUN as if it has more capacity than it actually has, to avoid fragmentatation and then expand it when it is needed.
  e.g. if you assign a 1TB LUN to a server, and then you need to expand the LUN due to lack of space, if you have space next to the already assigned TB you can avoid fragmentation. Besides, if you can put data in only 1TB instead of 2TB (even if you present the volume as if it had 2TB), internal fragmentation may happen only inside that TB, and later on you can expand the volume up to the reserved 2TB.
  Besides, available space may *decrease* over time, mostly due to snapshots (discussed later on).

- ◇ LUNs may perform **deduplication** (delete sequence of blocks that are equivalent/redundant, and exploiting indexes to retrieve duplicated data) to optimize storage performance. It is very useful in document-rich file systems, since people tend to copy a document multiple times.
- ◇ LUNs may perform **compression** to reduce the size of the data, and to increase the performance. It may be lossy or lossless. Its major downside is that it is computationally expensive, since the data must decompressed before using it. On the other hand, allows to spare bandwidth by sending compressed data, which we know to be critical.
  Searching in compressed data is not trivial, but there are tools to do it, such as the `FM-index`.
- ◇ LUNs may create **snapshots**, *"point-in-time"* copy of current data state, to save the differences between the current state of the data and the previous state of the data. This allows to recover the overwritten data in case of a failure, but it also takes up space.
  Snapshots older than a week are usually deleted, since they are not needed anymore.

Provisioning and Capacities

LUNs may be created from
- ◇ A RAID set (traditional approach); suited for application that require predictable performance
- ◇ A storage pool (modern approach); suited for application that require flexibility and scalability, and that tolerate performance variations.

Both of these approaches have different capacities:
- ◇ Row capacity: the total capacity of the LUN
- ◇ Usable capacity: the capacity that is available to the server
- ◇ Provisioned capacity: the capacity that is actually used by the server

5.4.3 Object based - S3

S3 stands for *Simple Storage Service*, and is a service that is used to store file data in the form of objects based on the content and other attributes of the data rather than the name and location of the file. The additional metadata (size, date, ownership...) or attributes (retention, access pattern...) enable optimized search, retention and deletion of objects.
A flat, non-hierarchical address space to store data provides the flexibility to *scale massively*.
S3 is leveraged to provide Storage as Service.

5.4.4 Big Data - HDFS

HDFS stands for *Hadoop Distributed File System*, and is a distributed file system that is used to store large amounts of data across multiple servers. A `map/reduce` algorithm is applied on the data, and then results are collected and summarized.
It exploits good forms of parallelism to run efficiently the algorithm

5.4.5 Unified - Unified Storage

Unified Storage or multi-protocol storage has emerged as solution that consolidates block, file and object storage into a single storage platform. It supports multiple protocols, such as NFS, SMB, iSCSI, FC, REST and SOAP.

> **iSCSI and its death**
>
> SCSI (Small Computer System Interface) was invented in 1979 for chaining drives through a bus (used for e.g. in fibre channels). The controller was so smart to allow the the drive to share the flat cable as a bus.
> Over the time some variants were invented, but the basic idea is the same. One example is iSCSI: *Internet Small Computer Systems Interface*, an IP-based storage networking standard for linking data storage facilities. It provides block-level access to storage devices by carrying SCSI commands over a TCP/IP network. The protocol died when SSD were introduced, since the latency was too high when communicating over the network.
>
> The key idea behind SCSI was for mutiple drives to share the same physical flat cable.
> It had been "deprecated" in favor of `NVMe`, but it is still used today, because it is very reliable.

5.4.6 Synchronization Software and its Price

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 a non-physical entity, increasingly more value, perhaps even too much.

5.5 Hyperconverged Infrastructure

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**.

5.5.1 HCI solutions

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.

"Adding servers adds capacity"

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. `read` operations are always performed locally on local drives; `write` operations instead sometimes require to retrieve a remote piece of data. A copy on the local server storage is kept, but the server needs to wait for the acknowledgment of the remote server in order to keep updated replicas of written data in other nodes.

Riak

Riak is a distributed database that is used to store data in multiple locations.
Recently it has been recognized that using general purpose hardware is no longer a feasible option.

5.6 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.

Chapter 6

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:

1. **Rack** servers: they are the most common type of servers, and are used for general purpose applications.
2. **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.

"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.

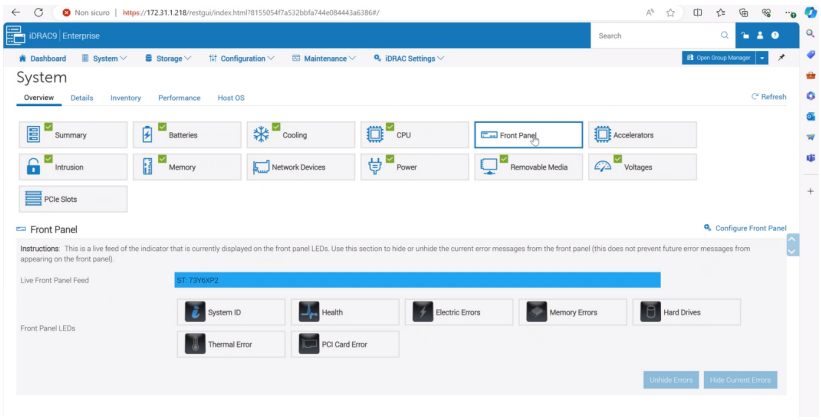

Figure 6.1: DEMO Interface displayed by prof cisternino in class
It is possible to remotely control and check the server's status.

CPU Affinity of a RAM bank indicates which CPU is connected to that bank.
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.

[1] May have other names, but this is a common one, used by `Supermicro`

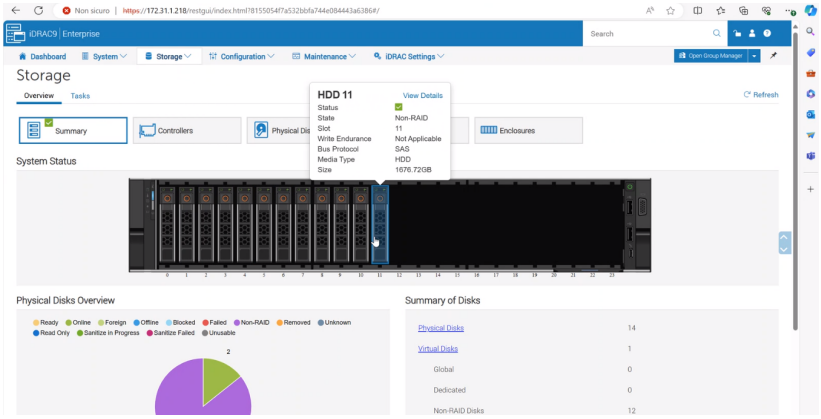

Figure 6.2: Monitoring storage
SAS bus protocol is used for storage devices.

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.

No phyisical security means no security at all

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.

6.1 Knights Landing and high performance computing

Knights Landing is an old multi —up to 72— core architecture is 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.

6.2 Rings

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. Nowdays 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.

6.3 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.

TOP500 is a list of the 500 most powerful supercomputers in the world. It is updated twice a year, and is used to compare the performance of supercomputers.

Chapter 7

Virtualization

Virtualization consists in virtualizing hardware resources, such as CPU, memory, storage, and network interfaces. This allows to run multiple operating systems on the same physical machine, which is called *host*.
It is not equal to *emulation* which consists in simulating hardware, giving the illusion that you are in another system, and is much slower than virtualization, since each instruction in the emulated system gets translated into up to —possibly— thousands of instructions in the host.

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 16 rings.

There are two kinds of virtualization systems:

1. VirtualPC (Microsoft), VirtualBox (Oracle), VMware Workstation (VMware), Parallels (Apple) : these are *desktop virtualization systems*, which are used to run multiple operating systems on the same physical machine. These solutions aim to provide "interactive" computers, with a GUI, peripheral support, etc.
2. VMware ESXi, Microsoft Hyper-V, KVM, Xen : these are *server virtualization systems* (**HyperVisors**), which are used to run multiple servers, typically GUI-less, on the same physical machine.
   Hypervisors introduce a **crucial** piece of software called **Virtual Switch**, which is responsible for managing the network of the virtual machines. The virtual switch's uplink is the host's physical network interface.

Similarly to snapshots in storage systems, there are **checkpoints**, which are used to save the state of a virtual machine at a certain point in time. This is useful to revert to a previous state in case of problems.

7.1 Network

HyperVisors introduce a **crucial** piece of software called **Virtual Switch**, which is responsible for managing the network of the virtual machines.
VMware is the leader in virtualization, but lately they have been changing pricings and licensing, which has made some customers unhappy.

Broadcom is a chip manufacturer, and we might end up with virtualization software already inside the chip.
VMware virtual switch is called **vSwitch**. It is a software-based switch that is responsible for managing the network of the virtual machines.
Every network interface of a virtual machine has its own MAC address, and may be connected to a vSwitch.
An hypervisor may handle multiple vSwitches.
From a network point of view, a virtual machine is just like a physical machine, assuming that the network card is in **promiscuous** mode, it can see all the traffic that is going through the vSwitch.

7.2 Live Migration

Hypervisors provide also the migration of virtual machines from one host to another, which is called **vMotion** in VMware. This is useful for load balancing, maintenance, etc. In Windows Hyper-V, this primitive is called **Live Migration**.
The migration is performed *without any service interruption*, only some degradation in performance and network latency. This also allows to move virtual machines from one host to another in case of hardware failure or phyisical maintenance. Besides, by redunding VMs we may also live switch from an older to a newer version of the software, without users noticing.

Live migration can performed like a context switch, by saving the state of the virtual machine and restoring it on the other host. This is possible because the virtual machine is not aware of the underlying hardware, and the hypervisor is responsible for managing the hardware resources.
Assuming that the disk is shared, the migration is performed like so:

1. The memory (and the registries) of the virtual machine is copied to the other host
2. If the copied memory is sufficient, the new VM starts to run on the other host
3. When data from the older memory host is requested, the virtual machine is paused, and the memory is copied again to the other host
4. Once the VM has been totally transferred, memory and other components belonging to the old VM are freed

vSwitches are also migrated, so that the network configuration is preserved. The old vSwitch may communicate with the new one, and if needed forward packets, until ARP tables are updated.

7.2.1 Replication

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.

Chapter 8

Containers

A VM is better than a process because it provides **isolation**: a typical problem in cybersec is that if an attacker cracks a process, he may access the whole system; with a VM, the attacker can only access the VM, not the host. However, a VM introduces overhead due both to the hypervisor and to the OS (cache, kernel, storage management...).
The idea is to tell a process that a process that its root is a subdirectory of the host's root, resulting in a strong isolation.
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.
Docker has become the de facto standard for containers, but there are other solutions like **LXC** (Linux Containers) and **rkt**. A Dockerfile contains the information to build a container.
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.
Docker containers are processes, and only see a portion of the host's filesystem.
Inside a Dockerfile you may create temporary containers by exploiting multiple images and lastly build the resulting slimmed image.

8.1 Upgrading a system

Typical procedure when a system needs to be updated: create first a snapshot before updating the containers (especially if there are updates related to the database), then update, run back the containers, check if everything works, and then delete the snapshot. If anything goes wrong, the snapshot will help to roll back.

8.2 Docker compose

Docker compose is a tool to define and run multi-container Docker applications. It uses a YAML file to configure the application's services, networks and volumes. With a single command, you can create and start all the services from your configuration.
Kubernetes is a more advanced tool for container orchestration, way more complex than Docker compose, it allows to manage thousands of containers, providing fundamental scalability features.

8.3 Docker security

An attack is to put the machine under heavy workload and observe from the container the performance to infer what other processes may be. This is called **side channel attack**.
Google has thousands of VMs and each runs a *container for each query*.

Chapter 9

Cloud

Cloud came out as a business model, not as a technlogy. It was needed to handle peak of requests and to allow scalability, without oversizing Infrastructures.
e.g. Amazon needs to handle way more requests on Christmas than on a normal day.
So, Cloud was a mean to reduce the cost of the ICT infrastructure.
When you program for the cloud, the consumer don't know where the process will be executed or where the data will be stored.
Resource pooling is the key concept of Cloud. It means that the services are provided to users using a multi-tenant model, with physical and virtual resources being dynamically allocated and deallocated according to the demand.
Cloud was needed also to provide rapid **elasticity**, meaning that capabilities may be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the customer it means that the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
Resource measurement is another fundamental pillar of Cloud: since the Cloud model revolves around pricing and resource consumption it's imperative to monitor and measure the usage of resources.
Benefits of Cloud:

- **Business agility**
  - Quick resource provisioning
  - Facilitates innovation
  - Reduces time to market
- **Reduces IT costs**
  - Reduces up-front capital expenditure (CAPEX)
  - Improves resource utilization
  - Reduces operational expenditure (OPEX)
- **High availability**
  - Ensure resource availability based on customer's requirements
    In RAI, prof. Cisternino experienced people complaining because their servers' CPUs were running at 98% of their capability, and they wanted to exploit also the remaining 2%, because "they paid for it".
  - Enables fault tolerance
    Recall active-active, active-passive, etc. configurations.
- **Business continuity**
- **Flexible Scaling**
- **Flexibility of Access**
- **Application Dev and Testing**
- **Simplified infrastructure management**
- **Increased collaboration**
- **Masked complexity**

Cloud has the magic power of decoupling the software from the hardware.
Disadvantages of Cloud:

- Vendor lock-in
- Privacy
- Your software depends on someone else
- Legislation is complicated
  In EU public administration data, must be stored in the EU.
- Availability depends on SLA, which may be written in a tricky way

9.1 Cloud Service Models

- **IaaS** (Infrastructure as a Service)
  - Provides virtualized computing resources over the Internet

  - Examples: Amazon EC2, Google Compute Engine, Microsoft Azure
- ◇ **PaaS** (Platform as a Service)
  - Provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure
  - Examples: Google App Engine, Microsoft Azure, Heroku
- ◇ **SaaS** (Software as a Service)
  - Provides software applications over the Internet
  - Examples: Google Apps, Microsoft Office 365, Salesforce

9.2 Cloud Deployment Models

- ◇ **Public Cloud**
  - Owned and operated by third-party cloud service providers
  - Deliver computing resources over the Internet
  - Examples: Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform

  It does **not** mean that the data is public. It means that the cloud services are accessible to the public.
- ◇ **Private Cloud**
  - Operated solely for a single organization
  - Managed by the organization or a third party
  - *On-premise* or *off-premise*

  It does **not** mean that the data is private. It means that the cloud services are accessible only to the organization e.g. *UniPi*.
- ◇ **Hybrid Cloud**
  - Composition of two or more clouds (private, community, or public) that remain unique entities but are bound together, and resources may be moved from one cloud to another (with some performance cost obviously)
  - By standardized or proprietary technology that enables data and application portability
- ◇ **Community Cloud**
  - Shared infrastructure for specific community
  - Managed by organizations or third party
  - On-premises or off-premises

9.3 Cloud as a Layered architecture - Cisternino's POV

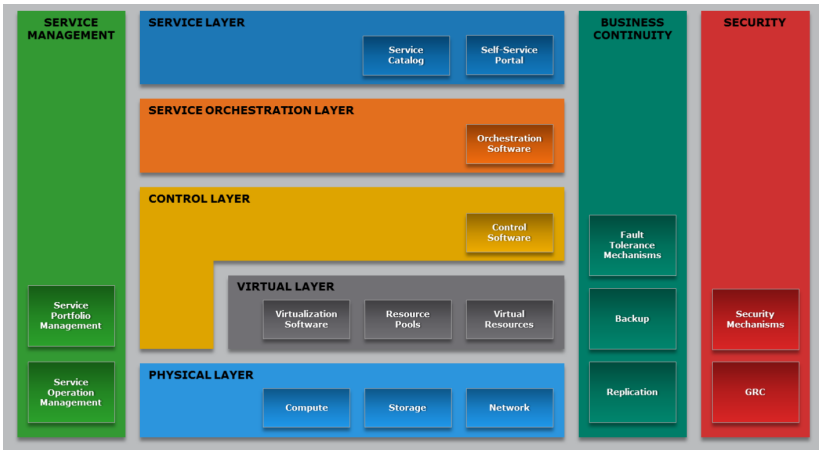

Figure 9.1: Cloud infrastructure as a layered architecture

In the last lectures of the Course, prof. Cisternino introduced the Cloud as a layered architecture, where each layer is responsible for a specific task, but went deeper only into the following topics:

- ◇ Software for control layer resource management
- ◇ Workflows: formalism to describe orchestration processes
- ◇ RPO and RTO key metrics
- ◇ The first two pages of Security
- ◇ Zero Trust Architecture
- ◇ Service Level Agreement
- ◇ Incident and Problem management

9.4 Physical Layer

Comprises **compute**, **storage** and **network** resources and executes both provider and consumer software. Compute systems are offered in the form of virtual machines.
Storage and access methods (e.g. file-based, unified, ...) have been exhaustively discussed in previous chapters.
For what concerns the network communications, they may be:

- ◇ **Compute-to-Compute** (East-West?[1]) : typically uses IP-based protocols
- ◇ **Compute-to-Storage** : typically a SAN exploiting depending on the fabric and architecture SCSI, iSCSI, FC, FCoE, NFS, SMB, etc.
  - FC SAN: Fibre Channel provinding up to 16Gbps bandwidth, and block-level access to storage
  - IP SAN: SAN using IP protocol and iSCSI or FCIP on top of it. Both encapsulate, respectively, SCSI and FC frames into IP packets
  - FCoE SAN: Fibre Channel over Ethernet
- ◇ **inner-cloud**: A different cloud interconnected to another one by WAN.

9.5 Virtual Layer

Abstract physical resources, including storage and network, and makes them appear as virtual resources; enables a single hardware resource to support multiple concurrent instances of systems or multiple hardware resources to support single instance of system.

Virtualization

Through **virtualization** a multi tenant environment is achieved, by running multiple —customer— organizations VMs on the same server; note however that this concept is not limited to VMs as a computing resource, but applies to storage and network as well. Virtual LANs, SANs and switches may be created.

Virtualization is composed by 3 entities:
- ◇ Virtualization Software
- ◇ Resource pool
- ◇ Virtual resources

Lastly, remember that Hypervisors, used to manage VMs and virtual resources, may be of two types:

- ◇ **bare-metal**: directly installed on the hardware. Performes better than hosted hypervisors, and typically are designed for enterprise datacenters.
- ◇ **hosted**: installed on top of an OS. Unlike bare-metal hypervisors, they do not have direct access to the hardware.

9.5.1 VMs network components

1. `vSwitch`: OSI Layer 2 switch that forwards traffic between VMs. May be internal or external, depending whether it connects solely VMs or if it is also attached to a physical NIC.
   A physical NIC already connected to a vSwitch cannot be attached to any other vSwitch.
2. `vNIC`: A vNIC connects a VM to a virtual switch, acting similarly to a physical NIC. Each vNCI has a unique MAC and IP address, and uses ethernet just as a physical NIC would.
3. `Uplink NIC`: an uplink NIC is a physical NIC connected to the uplink port of a vSwitch and functions an *Inter-switch link*. It is called uplink because it only provides a physical interface to connect a compute system to the network and is not addressable from the network, they are neither assigned an IP address nor are their built-in MAC addresses available to any compute system in the network. They simply forward the VM traffic between the VM network and the external physical network without modification.

9.5.2 Virtual Networks

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

[1] Perhaps not necessarily

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

9.6 Control Layer

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

Definition 9.1 (Control Layer) *"The control layer is important because it's the way pool 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

An *Element manager* has separate management tools for the three pillars listes, a *unified* instead exposes a unified (☺) management tool on top of them.

9.6.1 Resource Discovery

The control layer enables **unified manager** to learn about resources that are available for service deployment Provides visibility to each resource Enables to manage cloud infrastructure resources centrally

9.6.2 Resource Pooling and Provisioning

A unified manager at control layer allows to categorize in **grading pools** resources and identity pools based on predefined criteria. This helps creating variety of services decoupled from the actual hardware where they will run, but still providing choices to consumers on the type —and amount— of hardware they get (e.g. "0.5TB Flash, 4TB SATA, 1TB FC"). Multiple grade levels (e.g. "Gold", "Silver", "Bronze") may be defined for each type of pool, where costs/prices of resource pools differ depending on grade level.

Resource provisioning starts when a user requests a service.

When I create a VM, i can choose the template (e.g. "Windows 2016", "Ubuntu 18.04"), hardware, extensions, and lastly I will be asked to select a Host. At that point the system polls the control layer to see which hosts are available and which are not, and then rank them. Then I'll be asked for the chosen host the estimated workload on CPU (percentage), memory and disk. Set the host, the control software provides information regarding networking.

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

9.6.3 Control Software demo

Prof. Cisternino demonstrated a Control Software he uses for the UniPi Cloud, where he can see the resources available and the resources allocated to the VMs.

In the "VM and Services" view, the software allows to see the different Clouds associated to Polo 1, Polo 2, etc. and the resources allocated to each of them.
There is also another inner view to check VM networks and the actual hosts where the VMs are running.

He may even open a Powershell terminal on a VM.

This software allows him to manage about 1400 VMs.

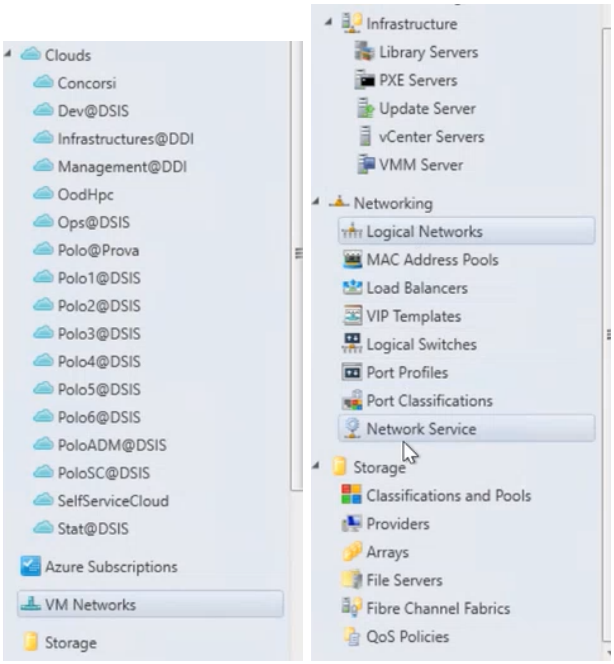

Figure 9.2: VM and Services view

9.7 Service Layer/Service Orchestration Layer

Definition 9.2 (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.

9.7.1 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.

Definition 9.3 (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****.*

UniPi~~.it~~ is a tenant for Microsoft Azure.

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

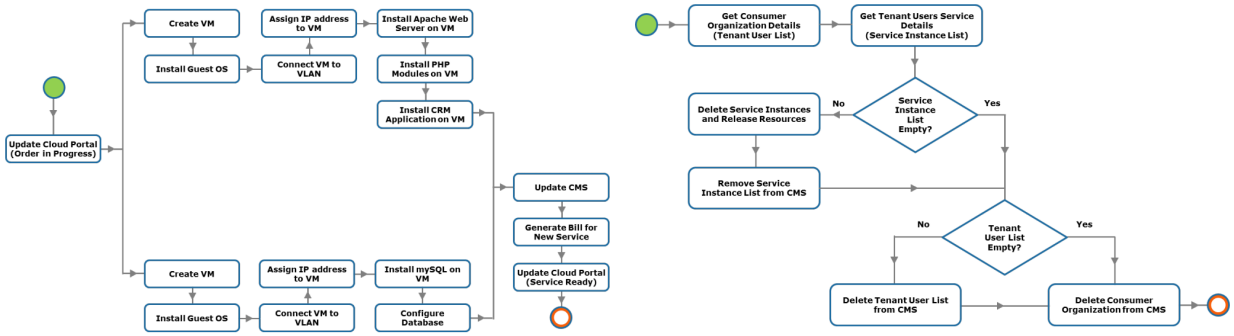

Figure 9.2: Service orchestration layer use cases. On the left, provisioning a CRM application, while on the right a tenant removal

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.

9.7.2 Deeper into Services

We may generalize the **lifecycle** of a service as it is depicted in Figure 9.3, 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

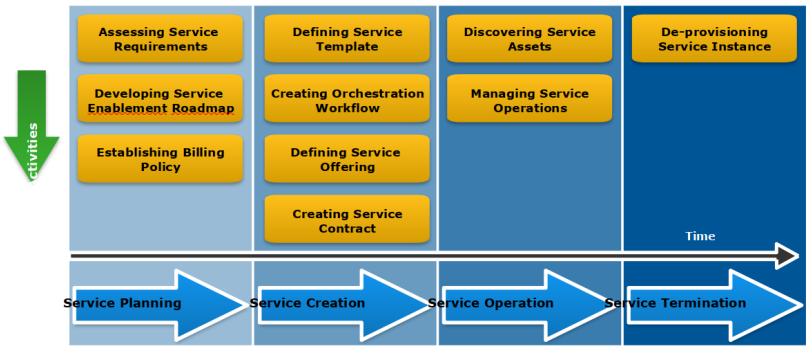

Figure 9.3: Service Lifecycle schema

Having defined what a Service is 9.2, we can now define the **Service Layer** as the layer that provides the services to the users.

Step 2.iv refers —also— to the Service Level Agreement, which basically is the legalese version of the set of resources that we are allocating for a user.
Aside from the SLA, other parameters are discussed such as pricing, termination of service and possibly other configuration aspects.

"If you are smart enough you may trick the user"

Amazon tricked a user by establishing a SLA that said that the user would have a 99.999% uptime, but they considered uptime even when the service was both not available and *not* requested by the user.

9.8 Business Continuity

Definition 9.4 (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 protection[2] solutions (*backup* and *replication*). It also involves **reactive** countermeasures, such as *disaster recovery*, to be invoked in the event of a service failure.

9.8.1 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 **backup**s (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 4.3.4, 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.

Also NICs may be aggregated

.

[2] Data is the most critical asset!

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.

9.8.2 VM Live Migration

Recall that in a VM live migration the entire **active state** of a VM is moved from one hypervisor to another. **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.

9.8.3 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.

9.9 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.

For cloud customers the key point is **trust**, which is provided by means of **visibility** and **control** on the data hosted.

[3] Point in time

9.9.1 Security threats in cloud environments

Threat	Description	Countermeasure
Data Leakage	Unauthorized access to data	Encryption Shredding unused data Multifactor auth
Data Loss	Data is lost due to a failure	Backup and replication
Account hijacking	Unauthorized access to an account	...
Insecure APIs	APIs are not secure	Restrict API access to authorized users
DoS	Denial of Service	Restriction on resource usage
Malicious insider	...	...
Abuse	Usage of cloud resources for malicious purposes	Establish agreement with customer on authorized activities
Loss of compliance	Provider does not adhere to laws and regulations	...
Loss of governance	Provider outsources its servers to a third party may lose control	...

Table 9.1: Security threats in cloud environments

9.9.2 Security Pillars and Datacenter Pillars

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".

We may break down the triad storage/compute/network of datacenters and see how security aspects get involved in each of them.

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**.

Authentication
- ◇ OAuth
- ◇ Multi-factor authentication
- ◇ Kerberos
- ◇ Challenge-response authentication
- ◇ OpenID

Authorization
Unix systems enforce a *default deny* policy, and users are added to groups to grant them permissions.
Windows has ACLs, which determine what can be done and what should be logged.
In general, role-based access control is the most common way to manage authorization.

Fun fact: today almost no internet service requires users to change password every 90 days, because it was found that it was counterproductive. People used to forget passwords and write them down onto notes that they would stick to the monitor or on the wall, completely breaking phyisical security.
To prevent VM exploitation, there are also **Virtual Machine Hardening** techniques.

Storage

- ◇ LUN masking
- ◇ Data Encryption
- ◇ Data Shredding

Network

IDSs and **IPS**s are security measures used to protect the network, and they are typically placed at the edge of the network, to prevent attacks from the outside; they are often combined with firewalls.
Sometimes they are not effective because intruders may remain silent for a long time, and then suddenly attack, or they may attack slowly in a way that the IDS/IPS does not recognize their actions as an attack, but rather as normal traffic or as a false positive in the worst case.
However, aside from intrusions, also tools for **network monitoring** and analysis are used, also for non-security purposes: we've already mentioned how important is monitoring everything inside a datacenter.
VPNs can extend a consumer's private network across the public network, and are used to secure the communication between the consumer and the cloud provider; the use of VPN may allow to avoid opening services to the public internet.
Other practices which somehow enforce network security are

- ◇ VLAN and VSAN
- ◇ Zoning
- ◇ iSNS discovery domain - same as FC zones
- ◇ Port binding - limiting the device that can connect to a specific switch port
- ◇ Fabric binding - only authorized switches maay join a fabric

9.9.3 Defense-in-depth or Layered Security

A common approach is to provide multiple onion-like layers of security, where each layer is independent from the others, and if one fails, the others are still there to protect the system.
The inner layer is typically defending the storage, which we know that *data* is the most valuable asset of an organization.

9.9.4 Zero Trust Architecture

The Zero Trust Architecture is a security model that requires strict identity verification for every person and device trying to access resources on a private network, regardless of whether they are sitting within or outside of the network perimeter.
Earlier on, security was approached as a multi-layered concept, and consisted of five levels, where the outermost was the perimeter, and the innermost was the storage.
The problem arose when people realized that the perimeter was not a good security measure, because once an attacker is inside the perimeter, it is game over. In other terms, security cannot be enforced based on the device itself and its location, we have no guarantee that is has not been compromised.

Zero Trust Architecture

Definition 9.7 (Zero Trust Architecture) *"Never trust, always verify"*

Almost every datacenter nowdays tends to follow the Zero Trust Architecture.

The ZTA consists of a set of principles called tenets, which are:

1. All data sources and computing services are considered *resources*, possible targets of an attack.
2. All communication is *secured*, regardless of the network location.
3. Access to resources is *granted on a per-session basis*, and is *limited to the minimum required.*
4. All access requests are *authenticated and authorized* before being granted.
5. All access requests are *monitored and logged*, as well as the security posture of all owned assets.
6. The enterprise collects as much information as possible about the current state of assets, network infrastructure, and communications, in order to improve its security posture.

Decisions are taken in the *Control Plane* through a *Policy Engine*, which enforces all the security policies of the organization. This *Policy Decision Point* (**PDP**) interacts with a *Policy Enforcement Point*, which checks whether your connection is trusted for a specific resource.

9.9.5 CIA/AAA Triads, plus other concepts

The **CIA Triad** is a widely used model for security policy development, which stands for:

- ◇ **Confidentiality**
  Ensures that information is only accessible to those who have the right to access it.

- ◇ **Integrity**
  Ensures that information is accurate and reliable.
  i.e. Unauthorized changes to data are not allowed.
- ◇ **Availability**
  Ensures that information is accessible when needed.

The **AAA Triad** instead stands for:

- ◇ **Authentication**
  Ensures that the user is who he claims to be.
- ◇ **Authorization**
  Ensures that the user has the right to access the resource.
- ◇ **Auditing**
  Ensures that the user's actions are logged.

TCB and Multi-tenancy

It is not advisable to make a single device responsible for enforcing security for the whole system, it is better to distribute such responsibility.
The TCB is the set of all hardware, firmware, and software components that are critical to the security of a computer system.
The TCB is the most critical part of the system, and it must be protected at all costs.

Velocity and Spray-and-Pray

Velocity-of-attack refers to a situation where an existing security threat in a cloud may spread rapidly and have large impact.
The majority of attacks are **spray-and-pray**, meaning that the attacker tries to exploit as many systems as possible, hoping that at least one of them is vulnerable.
Like a fisherman which goes out in the ocean, you throw a net, and check what you caught.

9.9.6 Data Privacy

Data privacy is a fundamental right, and it is regulated by the GDPR in the EU.
Even name and surname are considered personal data, and they must be protected.
It is not only a matter of law, but also of ethics. A name appearing in a list may affect the life of a person, what that will person will be allowed to do, and so on, even if no other information is provided.
Prof. Cisternino cited *Schindler's List*, to explain that even a list of names, with no other information such as address, date of birth, and so on, may decide what it will be of your life.

"Accountability" in GDPR

In GDPR appears the curious term *"accountable"*, i.e. TODO. Being able to produce all the documentation that proves that you are compliant with the GDPR, meaning that you did everything you should have done to protect the data.
Such term is different from "responsible". Accountability is about tracing back the actions, while responsibility is about the actions themselves.

"I'm accountable, I did everything I could to avoid such a bad situation to happen, but it happened anyway."

9.10 Service Management

Service portfolio management is the process of managing an organization's service portfolio to ensure that the services are aligned with the business goals and objectives.
Service operation management is the process of managing the operation of the services to ensure that the services are delivered as agreed in the service level agreements.
TCO (Total Cost of Ownership) is the total cost of owning and operating a service over its lifecycle.
ROI (Return on Investment) is the ratio of the net profit to the cost of the investment.

$$SALT = \textit{Service Asset Lifetime} \tag{9.1}$$

$$TCO = \sum_{t=1}^{t=SALT} \text{One-time costs} + \text{Recurring costs}(t) \tag{9.2}$$

$$ROI = \frac{\text{Gain from investment} - \text{Cost of investment}}{\text{Cost of investment}} \tag{9.3}$$

Chapter 10

Supercomputers

Recall that **TOP500** is the list of the 500 most powerful computer systems in the world. The list is updated twice a year and the first one was published in June 1993. The list is compiled by Hans Meuer of the University of Mannheim, Erich Strohmaier and Horst Simon of NERSC/Lawrence Berkeley National Laboratory, and Jack Dongarra of the University of Tennessee. The TOP500 project aims to provide a reliable basis for tracking and detecting trends in high-performance computing and to provide a basis for tracking the progress of the supercomputing industry.

10.1 Supercomputers in the TOP500 list

It is interesting that **Microsoft** has applied to have a supercomputer in the TOP500 list. It is ranked 11th in the list and is located in the United States. The supercomputer is called *Azure* and is operated by Microsoft. It has 2,596,016 cores and a performance of 27,580.0 TFlop/s. The supercomputer is based on the HPE Cray EX supercomputer and is used for commercial purposes.
Leonardo is the most powerful supercomputer in Europe and is located in Italy. It is ranked 7th in the TOP500 list and is operated by CINECA. The supercomputer has 14,000,000 cores and a performance of 32,800.0 TFlop/s. The supercomputer is based on the HPE Cray EX supercomputer and is used for research purposes.

"*Alps* is really really important" -prof. Cisternino

Alps is the most powerful supercomputer in the world and is located in Switzerland. It is ranked 1st in the TOP500 list and is operated by the Swiss National Supercomputing Centre. The supercomputer has 2,289,024 cores and a performance of 63,460.0 TFlop/s.

According to Cisternino it is of utmost importance because its CPU is designed by NVIDIA and it is the first supercomputer to use this technology. The CPU is called *Grace Hopper* and is based on the ARM architecture. The CPU is designed for high-performance computing and is used for research purposes.
It may represent a turning point in the CPU architecture choice, and ARM may become the new standard for high-performance computing, superseding x86.
Grace is capable of transferring up to 1Tbit/s of data between the CPU and the RAM, which is "quite a nice number".

10.1.1 Brexit and supercomputers

Almost one third of the computers in the list are in Europe, and in the top 10 there are 3 European supercomputers (?). Since most supercomputers were made in the UK, after the Brexit Europe focused on the need to have its own supercomputers.
This is why the *Leonardo* supercomputer was built in Italy.

Chapter 11

Exam

1. *Do you think that cabling may affect the PUE?*
   Yes, because the cabling may affect the airflow and the cooling system, which in turn may affect the PUE. But not only this, because part of such PUE may be hidden in the "compute" part of a system: fans inside the servers, when detecting a higher temperature, may increase their speed, (or lower the CPU performance) thus consuming more energy.

11.1 Domande mie

11.1.1 What is the standard voltage of incoming power in a data center? Where does the conversion from AC to DC happen?

11.1.2 Three-tier architecture: What does it mean that "provides active-passive redundancy which leads to inefficient east west traffic"?

Answer by Copilot: "The three-tier architecture provides active-passive redundancy, which leads to inefficient east-west traffic. This is because the traffic between servers in the same tier must go through the core layer, which can lead to congestion and increased latency. In contrast, a spine-leaf architecture provides active-active redundancy, allowing traffic to flow more efficiently between servers in the same tier."

11.1.3 LACP: even though the bandwidth is aggregated (i.e. $2 \times 25Gbps$), the single stream is still limited to the bandwidth of a single link (i.e. $25Gbps$), because the traffic goes only from one way to the other each time.

Answer by Copilot: "LACP (Link Aggregation Control Protocol) is a method of combining multiple physical links into a single logical link to increase bandwidth and provide redundancy. However, even though the bandwidth is aggregated (e.g., 2x25Gbps), a single stream is still limited to the bandwidth of a single link (e.g., 25Gbps) because the traffic is hashed based on the source and destination MAC addresses, IP addresses, or TCP/UDP ports. This means that traffic between the same source and destination will always use the same physical link, limiting the effective bandwidth of a single stream to that of a single link."

11.1.4 Spine-leaf: "with spine and leaf we introduce more hops". Weren't the necessary hops no more than one?

11.1.5 Why would a 15TB disk be better than a 27TB disk?

Assume the same performance, and the same price.
It would be preferrable because it would take less time to extract all the data from the disk[1], since it is smaller.
However, large capacity drives are used for *cold storage*, where the data is not accessed frequently, speed is not a priority, and even if the data is accessed, only a portion of the disk is needed at a time; in case of failure and thus needing to retrieve an entire backup, the time taken to retrieve the data is not a priority, since this —hopefully— happens only "once".

[1] i.e. taking advantage of the space provided

11.1.6 **Optane can perform 416 accesses in the same time needed by a mechanical hard drive to perform 1 access. It looks like the latency in this latter case is neglegtible. Someone may be tempted to reduce the size of read/write operations and perform multiple smaller ones, since "it's free". Why is this not a good idea or why is it?**

11.1.7 **SAN also enables performance optimization of the storage by performing deduplication (delete sequence of blocks that are equivalent). Why/How?**

11.1.8 **What is the difference between a SAN and a NAS?**

11.1.9 **"Storage capacity of a LUN can be dynamically expanded or reduced by means virtual storage provisioning, i.e. present a LUN as if it has more capacity than it actually has, to avoid fragmentatation and then expand it when it is needed.". Why does this avoid fragmentation?**

11.1.10 **Define row, usable and provisioned capacity**

11.1.11 **Cloud Phyisical Layer - "Compute systems are offered in the form of virtual machines". Isn't virtualizing a job of the control layer?**

11.1.12 **The hypervisor demonstrated by prof Cisternino is bare-metal or hosted?**

11.1.13 **"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." Why is this useful? Besides, also NICs may be aggregated, why is this useful?**

11.1.14 **What about physical security for what concerns storage in a DC?**

11.1.15 **Why in the CRM application workflow example two separate VMs are used?**

11.1.16 **How does scaling an application work in a cloud environment?**

Chapter 12

Questions of previous years

12.1 Spine and leaf VS traditional architecture

12.1.1 Question

Discuss the difference between spine and leaf fabric and the more traditional fabric architecture based on larger chassis. How bandwidth and latency are affected?

12.1.2 Solution

12.2 Spine and Leaf

Non modular, fixed switches are interconnected with some MLAG (Multi-chassis Link Aggregation). Loosely copuled form of aggregation: the two switches are independent and share some form of aggregation. Each leaf is connected to all the spines (if the leaf has 6 upwards ports, 2 are used to connect the two coupled switches in the leaf ,the others are used to the connection with the spine). At least 2 spines for redundancy. The spines are not connected each other. LACP protocol allowing to bind multiple links to a single conceptual link (link aggregation, active-active).
over-subscription the links to the spine should be able to sustain the trafic coming from all the links below. This is not a problem for EW trafic between servers attached to the same switch (because the link to the spine is not affected).
Pros:

- resilient
- active-active
- can be updated while the system is running

It became popular after 10 GBps; before it was difficult to use it with 4/8/16 ports per server. Different VLANs are used.
Traditional Chassis
Tipically two modular chassis connected by two links (STP) in atcive-passive (the second chassis goes up only when the first isn't working). The ratio between the number of ports and the bandwidth is completely different from spine and leaf. Link aggregation is possible but it's not convenient.
Pros:

- room for growing
- protection on the investment
- share power
- pay only once and just add line cards
- simplifying the power supply cabling

Today is not so much used because it's difficult to design a backplane offering terabits.

- **Capex and Opex reasons**: in active-passive I use only half of the bandwidth I'm paying for.
- **latency issues**: with STP when a link goes down it can take up to seconds to activate the other link.

Latency
With spine and leaf we introduce more hops, so more latency, than the chassis approach. The solution for this problem is using as a base of the spine a huge switch (256 ports) which actually acts as a chassis, in order to reduce the number of hops and latency.
Bandwidth
To enlarge the bandwidth in a spine and leaf architecture we need only to add a new spine and to connect to all leaves. With the chassis approach we can add bandwidth adding new line cards (new switches) to the chassis, provided that there are free slots in the chassis.
In the spine and leaf arch we can upgrade a spine reducing *(???)*1 the bandwidth, but still without disrupting the connectivity. In the traditional chassis an upgrade degrades the bandwidth => TODO: verify.

How much Bandwidth?

How much can the bandwidth be increased by adding new switches? If a server has only one NIC, isn't the bandwidth limited to the NIC? Adding switches may mean adding links for east-west traffic, but which is the upper bound?
Actually the max number of links is the number of arcs in a complete graph, which is $n(n-1)/2$ where n is the number of nodes.
Pretty unreachable, but also quite messy to manage as fabric.

12.3 2) Orchestration layer

12.3.1 Question

What actions can take the orchestration layer of a cloud system, and based on what information, in order to decide how many web server istances should be used to serve a Web system?

12.3.2 Solution

Assuming the DB is distributed and has infinite capacity, because tipically the bottleneck is the Web Server
An orchestrator can:

- create new VM running the WS, getting a new IP and talking to the **Load Balancer**
- delete a VM
- save a VM (freezing it)
- increase memory
- etc..
- *Not migration! (right?)*

Based on:

- average response time
- available memory in the —VM running the— WS?
- latency on web requests (if it goes beyond a threshold spawn another service)
- number of connections (requests)
- CPU usage
- *What about the pricing plan of the customer?*

12.4 3) Datacenter architecture

12.4.1 Question

Discuss a datacenter architecture made of 10 racks. Assuming a power distribution of 15 KW/ rack.

12.4.2 Solution 1

Use an in row cooling approach trying to reduce the rows to be cooled. Do not forget to mention the PDU and the UPS. (2 plugs per rack 32A each).
Some claculations:

1. Calculate the amount of current per rack:
   - $15000W/380V \simeq 40A$ per rack

   Why 380V? Standard Voltage in DCs instead of 230V?
2. Each rack has 40A, so assuming that is contains 42 servers we have:
   - $380V \cdot (40A/42) \simeq 360W$ per server (slightly less than 300 are required for the sole CPUs)

   Correct to say this is insufficient? We might need less units per rack?
3. Calculate the amount of current on the PDU:
   - 40A*10 = 400A for the racks
   - assuming a PUE of 1.2 and knowing that

$$PUE = \frac{total\ \ current}{compute\ \ current}$$

   - calculate the toal current
     - total current = 1.2 * compute current = 1.2*400 = 480 A on the PDU, that must be spread between racks and cooling systems.

1. Dimension the UPS:
   - Assume that in case of PDU issues you want to keep alive olny half racks, you can buy a UPS capable of generating 240A

NB. We have not considered the power factor, which is a number equal to 1.0 or less. Reactance, obtainied by converting AC in DC, reduces the useful power (watts) available from the apparent power. The ratio of these two numbers is called the power factor (PF).
We haven't talked about Power factor right?

12.4.3 Solution 2

t1 and t2 are AC/DC transformers
ATS: Automatic Transfer Switch
15 KW/Rack x 10 Racks = 150 KW to deliver towards our DC
380 Volts x 150 KW = 400 Ampere (at least)
Assuming a PUE of 1.2, we need:
400 Ampere x 1.2 = 480 Ampere (including cooling and AC/DC power loss)
So, every couple UPS/PDU must manage 480 Ampere at full load.
According to Facebook, by eliminating centralized UPS/PDUs you can achieve a total loss of 7%.

Figure 12.1: Solution 2 schema

12.5 4) SAN VS hyperconverged architecture

12.5.1 Question

A service requires a sustained throughput towards the storage of 15 GB/s. Would you recomment using a SAN architecture or a hyperconverged one.

12.5.2 Solution

- 15 GB is the max bandwidth of a PCI express bus with 16 lanes.
- 100 Gbps bandwidth of a single —FC?— link (even if internally is 4*25 Gbps).
- 15 GBps * 8 = 120 Gbps
- The PCI has some overhead, so it's bandwidth is not fully 15 GB.
- 15 GB = 54 TB/hour = 1 PB/day = half exaB / year. I have also to consider where am Is going to store this data, not focusing only on the bandwidth.
- 8/ 10 TB mechanical drive capacity.
- SATA SSD has 500 MB bandwidth.
  What about NVMe?
- With 15 GB/sec, 4 fiber channel are enough.

SAN area network (recap)
uSCSI internet protocol (SCSI on fiber) allows to mount blocks/disks.
Block-based access: you mount a chunk of bytes seen as a drive.
LUN (Logical UNits), can be replicated, compression can be used, it can be overbooked.
Servers and drives are separated, drives are pooled togheter.
SAN today is failing because the bandwidth of the drives saturates the links, making it impossible to pool a lot of drives.
NAS
It uses istead a network file system protocol to access the pooled resources (SIFS, NFS). We access files not blocks.
NAS gives the file system, with SAN I decide the FS.
Security in SAN is bounded to the compute OS, which decide the authentication domain.
Instead NAS has the responsibility of the security and the filesystem abstraction(Active Directory and NFS security).
Both SAN and NAS separate the storage from the compute. Configure one for all the storage (backup, compression. . .) and look at it as blocks or files.
This architecture is failing because of the throughput of the drive (very fast) that saturates the link.
HCI (hyperconverged)
Before we talked about three independent units: compute, storage and network. With HCI instead we have boxes (servers) with a little bit of network, drive and compute.
It's not true that the compute and the drive are completely unrelated and can be completely separated: also the CPUs have their own limits in data processing even if large (risk to waste resources).
HCI by Nutanix allows to simply add a bit of storage, a bit of compute and a bit of network by buying a server. You pay as you grow and lower the risk, since you don't have to make capacity planning.
Discussion
The choice depends also on the kind of data I assume to process (assume at least one: sensors, bank financial data . . .). For example HCI is not convenient if a want to do archiving because I pay for extra unused CPU.

It's not enough to say: I take 5 big drives, because their bandwidth can be a bottleneck.
SAN could be the good solution because it's cheaper. SAN can be used with **tiering**: in the first layer I keep SSD "buffers", in the second layer mechanical drives. If I keep a buffer of 1TB I'll have 6 minutes to copy down the buffered data to the mech drives.
Assuming 24 Gbps of incoming bandwidth and 1 TB of SSD buffer.
This way you would need 30 ($= \frac{15GB/s}{500MB/s}$) SSD drives to avoid bottlenecking the incoming 15GBps bandwidth.
24 Gbps = 3 GBps –> 1000 Gb /3 = 330 s to saturate the buffer.
Netxt to the buffer there are mech drives (130/150 MBps)
I write to the SSD 3000 MBps but I copy to the drive (assuming just 1) 150 MBps. So the incoming bandwidth in the buffer is 3000 -150 = 2850 MB/s.
With one mech drive I will saturate the disk in 1000 GB / 2.8 GBps = 360 s = 6 min
If I consider the text of the exercise, in particular 'towards', as in the sense of "only writing", imagining to have to almost only archive data and read only from time to time, I can actually consider SAN, because if I go hyperconverged I am paying also for the CPU which might be unused. If I instead have a balance between r/w and want a good throughput for both operations, or I have a peek and then a flatter period of time with few action, then I might choose better going hyperconverged.
What should I look for..

- Capex Opex
- Resilience
- Bandwidth (network, drives)
- etc...

12.6 5) Dimension a hyperconverged system

12.6.1 Question

A service requires a sustained throughput towards the storage of 15 GB/s. How would you dimension an hyperconverged system to ensure it works properly?

12.6.2 Solution

Look first at the network (fabric is the glue of the infrastructure). Can't have 100 GBps straight to the server because of spine and leaf, so I have to consider the idea of distribution (hyperconverged). Why not? 100Gbps is the max bandwidth of a single link, but we can have multiple links, right? Could I add more links/NICs?
Recap that:

- just 1 or 2 ports of 100Gbps are enough to saturate the PCIe.
- not good to have 100Gbps for each node cause I'm overloading that single node while HCI is distributed

First I have to choose the Ethernet bandwidth between (10-25-50-100-400), considering that 400 Gbps is achievable only on the spine, and not on the leaves.
Better 10 Gbps or 25Gbps depending on CAPEX.
With spine and leaf I have 50 Gbps cause I double (active-active).
Some calculations
We have 15 GB/s incoming bandwidth –> 15 * 8 = 120 Gbps
We first dimension a spine and leaf architecture to sustain this bandwidth value.
We have a couple of options:

- 10 Gbps per server (hyperconverged node)
- 25 Gbps per server (we choose this)

To cover 120 Gbps we need at least 5 nodes –> 5*25 = 125 Gbps
We could also add more (up to 8-10) nodes to have redundancy and efficiency, but we will consider 5 in the calculations
Every HCI node will have some SSD (as buffer) and some mechanical drives.
Since we have 120 Gbps totally each node will receive 120 / 5 = 24 Gbps storage bandwidth
This is ok since the link to the node is 25Gbps (even if we have active-active configuration so the actual bandwidth is 50 Gbps)
Now we must consider the number of drives in each node. The drive throughput must sustain the incoming bandwidth of 24Gbps to avoid data loss. We know that SSD drives have a bandwidth of 500 MBps, so half a GB.
So 24 Gbps / 8 = 3 GBps

- $\#disks \cdot (\frac{1}{2} GBps) = 3GBps \implies \#disks = 6$

Remember that bandwidth is not fully used because of some overhead (e.g. to connect two spine nodes together).
So we have 5 leaf switches, each providing 25Gbps to the server, and each server has 6 disks, each providing 500MBps. Using 5 servers we have 125Gbps of storage bandwidth, which is enough to sustain the 120Gbps of incoming bandwidth.
What has to done at the spine layer? Which type of protocols and connectors must be used in order to achieve the abovementioned bandwidth?

12.7 Other questions

12.8 @megantosh

My exam was a variation of the above questions:
Design a data center that should contain 40 Racks, consuming 15 kW each. Discuss all necessary considerations, e.g. Power Distribution, Cabling, Cooling.
drawing a scheme with the components like PDU etc. was appreciated. Discussing firefighting, cooling using natural resources (water from ocean etc.). Show that you can do the Math: 15000 Watt = 380 V * A * cos fi where cos fi is the heat dissemination happening from conversion of AC into DC current
1024 Servers need to be connected with any of the following switch options: 48x25Gbps East/West Traffic with 6 x 100 Gbps for North/South (and two other configs to choose from). Oversubcription level can be up to 1/6. Which network Topology would you choose? Discuss its pros and cons.

12.8.1 Solution 1

I gave a couple but he was more interested to see one discussed in through detail. So not to recite theory but to be able to apply the knowledge from the section above.
For 1024, 48x25, oversubscr 1/6. I went for spine/leaf model and he wanted to know how many switches would be required in that case (do not forget redudancy causes doubling + two links of the 6 will be gone for connecting the two spines together)

12.8.2 Solution 2

E/W : 48 x 25 = 1200 Gbps
N/S : 6 x 100 = 600 Gbps
1200 / 600 = 2:1 Oversubcription (Good) Why is this good?
How many switches should I buy?
1024/48 = 21,333 = 22 switches (leaves) at least Why 22? I don't get the math here. Are we dividing the number of servers by the number of ports?
I don't have enough space to put every switch in a single rack (tor), can I re-organize the infrastructure in order to save space? What's tor? What's the advantage of putting switches in a tor?
Yes, you can "merge" two switches phisically by switch aggregation and put them in a rack (tor). To aggregate two switches, we need to connect them with 2 cables (for resiliency). This means that we lost 2 ports on each side, so, a new oversubcription ratio should be computed.
If we imagine two of the given switches put together, we get:
E/W : (2 x 48 ports) x 25 Gbps = 96 ports x 25 Gbps = 2400 Gbps
N/S : [2 x (6 - 2) ports] x 100 Gbps = 8 ports x 100 Gbps = 800 Gbps
2400 / 680 = 3:1 Oversubcription (Still good)
How does live migration of a VM happen and would you prefer to do it over HCI or SAN?
as long as no detail is provided, you are allowed to make your own assumptions.
I said both should be effectively the same given that bandwidth would not be blocked and that we are using latest/-fastest technology. Crucial part of the VM Migration was (apart from copying config files, moving virtual registers and halting the old machine for a millisecond) is that copying the files happens in the background. If a file is required to complete a process and it has still not been migrated, the new VM goes and fetches it first from the old VM before it continues with copying any other file*
Wasn't this related to (RAM) memory areas more than "files"? Having a SAN allows to have a shared storage, so the VM can access the same data from both the old and the new machine. What about HCI? Should happen the same but inside the HCI "box", right?
Discuss the role (functions) of the orchestration layer. Give an example workflow. where does it lie in the cloud stack?
check the slides for sure, they are very helpful! A process I gave was provisioning an Alexa skill on AWS which requires building a Lambda function (an AWS service) and a Skill controller. I think he was happy to have a real-life example. Draw the workflow like a business process from the point of provisioning to the billing etc.

12.9 @giacomodeliberali

What is a virtual firewall? Where will you put it? And how many?
TODO: answer
"Virtual" firewall? Don't remember talking about that. However i would put one for each N/S link, probably none for E/W links, IDS/IPS there would be more appropriate, to detect lateral movements.

Chapter 13

About numbers

13.1 Current

- Watt = cos fi * A * V
- The Industrial current il 380 Volts, 3 phases
- 32 KW (consume of 10 appartments) datacenter is a small one
- 2 idle CPUs (14 cores each) consume 140 Watts
- Intel XEON (28 cores) 165 W on avarage
- lines coming from the generator to the UPS: 80 KW each, ~ 200 A; can be used for 6 racks each 32 A (I will not use the whole 32 A so I'll put more racks)
- rack PDU: 2 banks, 12 plugs each
  - 16 A each bank
  - 15 KW per rack
  - 42 servers per rack (42 U) => ~ 350 W x server (2 CPUs 140 W look above)

13.2 Fabric

- ethernet MTU 9 KB (jumbo frame)
- TCP/IP introduce 70-100 micro sec of latency
- Infiniband MTU 4 KB, latency 2 micro sec
- servers with 1 Gbps connection (not so high) connected to 48 ports switch => switch uplink 48 Gbps.
- 100 Gbps uppper bound nowadays for ethernet (using QSFP28, 4 * 25 Gbps)
- other bandwidths
  - 25 Gbps
  - 50 Gbps (2* 25)
  - 16 Gbps
  - 40 Gbps (4 *10)
- STP can take up to 30-50 sec to converge
- some switches up to 1 KWatt (years ago 200 W)
- **spine and leaf**
  - LACP links 2*25 Gbps
  - 1/3 ports upwards, 2/3 downward
  - 48 ports 10 Gbps (downward)
    - plus 6 ports 40 Gbps each (upward)
    - oversubscription. 48*10 / [(6-2)*40] = 3/1
  - 48 ports 25 Gbps each (downward)
    - 6 port 100 Gbps (upward)

13.3 Disk and Storage

- PCI express bus 15 GB/s = 120 Gbps
  (PCIe Transfer rate per lane increased exponentially since 2003)
- 4 drives can saturate a full PCIe bus and also a 100 Gbps link
- NVMe drives up to 11 GBps
- Intel Ruler up to 1 PB capacity
- SATA SSD 500 MB bandwidth
- SATA HDD 130 MB bandwidth
- nVME SSD 24 Gbps (3 GBps) bandwidth

- ◇ 4 fiber channel are enought with 15 GB/s
- ◇ SATA interface 16 Gbps
- ◇ SCSI interface 22.5 Gbps

Chapter 14

External resources

14.1 Real Use Cases

- Introducing data center fabric, the next-generation Facebook data center network
- Facebook - Designing a Very Efficient Data Center
- Google - Efficiency: How we do it

14.2 Open Source

In 2011 Facebook announced the Open Compute Project (OCP), an organization that shares designs of data center products among companies, including Facebook, IBM, Intel, Nokia, Google, Microsoft and many others.
Their mission is to design and enable the delivery of the most efficient server, storage and data center hardware designs for scalable computing.

14.3 Books & Guides

- Cisco - Design Zone - Design Guides
- Building a Modern Data Center
- IBM Data Center Networking

14.4 References

- https://tools.ietf.org/html/rfc4391
- Omni-Path - https://en.wikipedia.org/wiki/Omni-Path
- https://en.wikipedia.org/wiki/Remote_direct_memory_access
- https://www.arubacloud.com/infrastructures/italy-dc-it1.aspx
- https://en.wikipedia.org/wiki/Software-defined_networking
- https://en.wikipedia.org/wiki/Software-defined_storage
- https://en.wikipedia.org/wiki/Software-defined_data_center
- https://en.wikipedia.org/wiki/Spanning_Tree_Protocol#Rapid_Spanning_Tree_Protocol
- https://en.wikipedia.org/wiki/Multitier_architecture
- http://searchdatacenter.techtarget.com/definition/Leaf-spine
- https://en.wikipedia.org/wiki/Network-attached_storage
- https://en.wikipedia.org/wiki/Non-RAID_drive_architectures
- https://en.wikipedia.org/wiki/Fog_computing
- https://www.openfogconsortium.org
- https://en.wikipedia.org/wiki/Power_usage_effectiveness
- https://howdoesinternetwork.com/2015/what-is-a-non-blocking-switch
- https://en.wikipedia.org/wiki/Network_function_virtualization
- Drives performances - http://www.itc.unipi.it/index.php/2016/02/23/comparison-of-solid-state-drives-ssds
- Drives performances - http://www.itc.unipi.it/wp-content/uploads/2016/05/ITC-TR-02-16.pdf
- HCI - https://www.nutanix.com/hyperconverged-infrastructure/
- Spine and leaf - https://community.fs.com/blog/leaf-spine-with-fs-com-switches.html
- Spine and leaf - https://blog.westmonroepartners.com/a-beginners-guide-to-understanding-the-leaf-spine-n
- Power consumption - https://searchdatacenter.techtarget.com/answer/How-do-I-estimate-server-power-consu
- UPS dimension - https://searchdatacenter.techtarget.com/feature/How-do-I-figure-size-requirements-for-
- Automatic Transfer Switch - https://ethancbanks.com/2014/08/22/what-is-an-automatic-transfer-switch-power

- Securing a datacenter - https://www.youtube.com/watch?v=cLory3qLoY8
- Live migration of a VM in nutanix HCI - https://www.youtube.com/watch?v=LlArzoASFNc
- Live migration of a VM (Hyper-V and vMotion) - https://searchservervirtualization.techtarget.com/feature/Lets-get-this-straight-VM-live-migration