ICT Infrastructure - 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 a server units in a rack. Typically, for each PDU there is another one, providing redundacy and thus resilience/robustness.

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.

There are two **drawbacks**:

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

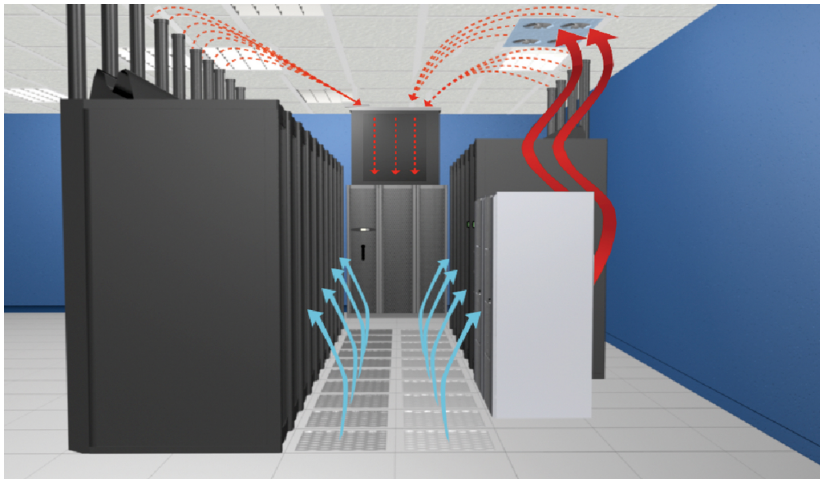

Figure 2.1: CRAC/CRAH cooling architecture

1. 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. 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.

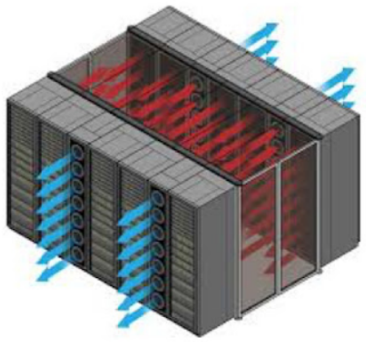

Figure 2.2: Inrow cooling architecture

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, *"Canť 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 provide much more. 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 $\sim$ 1000`Gbit/s`—, 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.

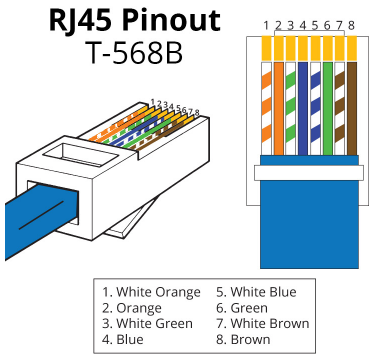

Figure 3.3: RJ45 - T568B

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.

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.4
- ◇ 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 a 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 not 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 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-mentioned traffic bottleneck.

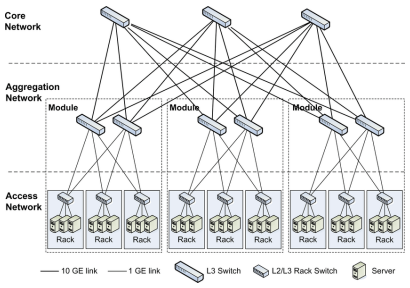

Figure 4.1: Three-tier architecture schema

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.

4.3.3 Spine Leaf architecture

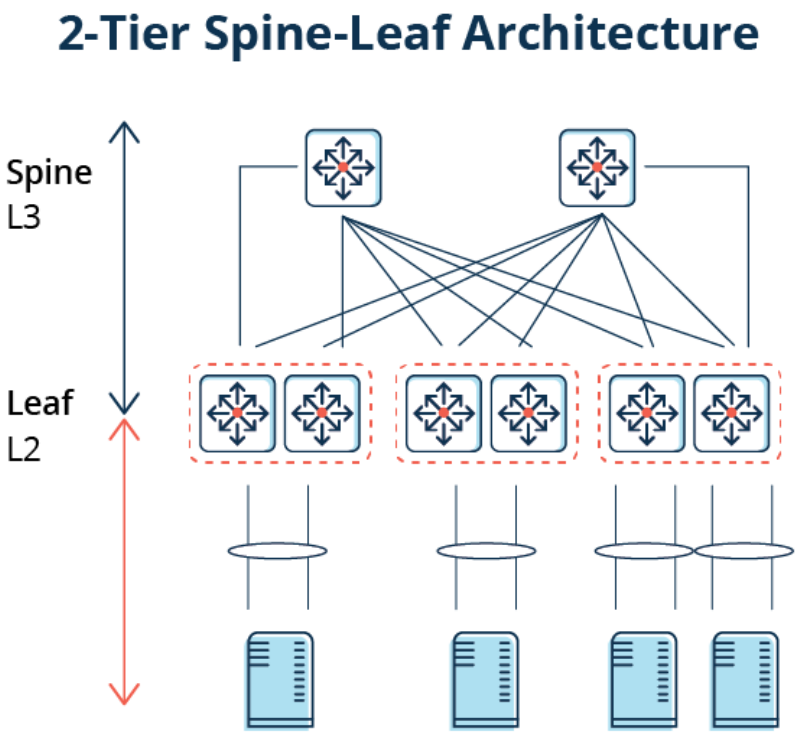

Figure 4.1: Spine-leaf architecture schema (from Arubanetworks.com)

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 one ☺) away from any destination, so latency is lower and predictable (fixed).

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 (because you can mix and match devices) with fixed size switches.
- ◇ Latency predictable: every host is distance one or two hops to each other host.
- ◇ Bandwidth control (it's possible to chose the proportion of NS and EW traffic) and overbooking (overbooking explained after).
- ◇ Active-active redundancy (because 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 switch, upgrade it and reboot it without compromising the network. Half of the bandwidth is lost in the process, but the twin switch keeps the connection alive.

A typical configuration of the ports and bandwidth of the leaves is:

- ◇ 1/3 going upwards and 2/3 going downwards
- ◇ 48 ports 10 Gbps each (downward - from leaves to racks)
  - plus 6 ports 40 Gbps each (upward - from leaves to spines)
- ◇ or (typical switch) 48 ports 25 each (downward)
  - plus 6 ports 100 each (upward)

Just a small remark: 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.

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 mod- ern 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.

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

4.3.4 Full fat tree

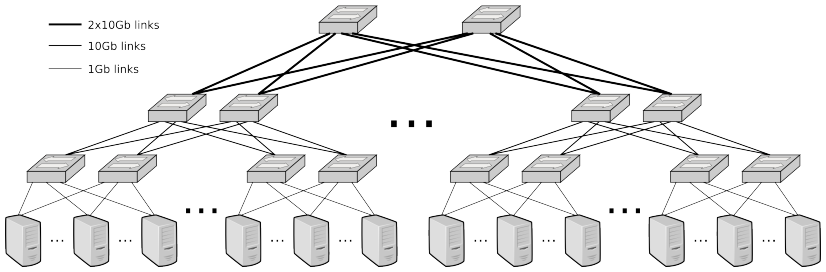

Figure 4.2: Full fat tree network topology schema

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.

4.4 Virtualization

With VLAN frames are extended by 4 bytes[1]. 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. 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.

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 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.

[1] 12 bits are reserved for the VLAN ID allowing up to 4094 ($4096 - 2$) logical partitions

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 *us*, 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

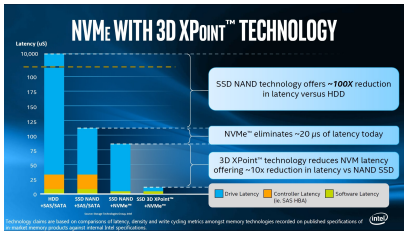

Figure 5.2: Storage types comparison
NVMe basically removes the orange part of the figure, which is the latency introduced by the controller, since it is a *controller-less protocol* and allows to access the storage directly from the PCIe bus.

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.

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

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 $\frac{TBW}{365 \times Years\ of\ Warranty \times capacity}$

.

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.

5.1.1 Tiering - Memory Hierarchy

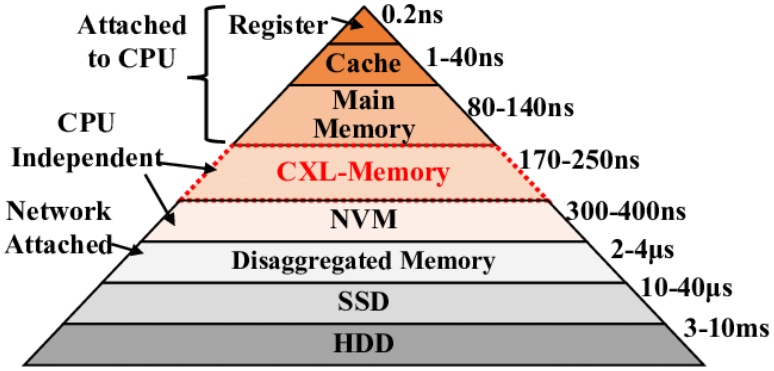

Figure 5.1: Memory tiering hierarchy

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.

5.1.2 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.
Fiber channel is the fabric dedicated to storage; the link coming from the storage ends up in the *HBA* (Host Bus Adapter) in the server.

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

5.1.3 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.

5.2 Redundancy and backup

5.2.1 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.

5.2.2 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.

5.3 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

5.3.1 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*.

5.3.2 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.
  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

5.3.3 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.

5.3.4 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

5.3.5 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.

5.3.6 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.

5.4 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**.

5.4.1 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 VM acts as a controller implementing 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.

5.5 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

"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.

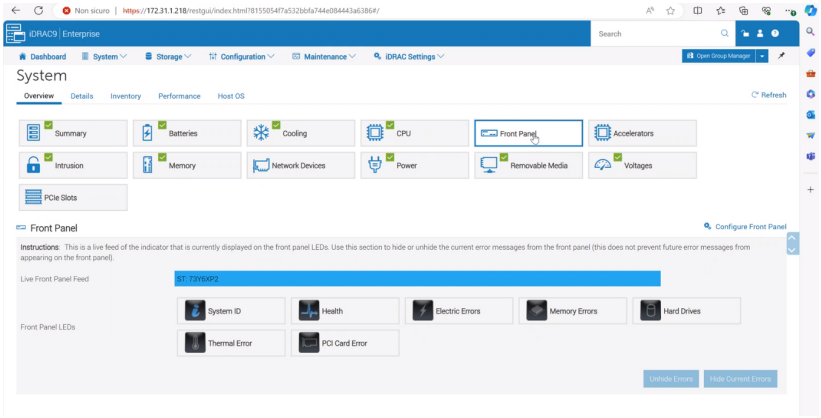

Figure 6.1: DEMO Interface displayed by prof cisternino in class
It is possible to remotely control and check the server's status.

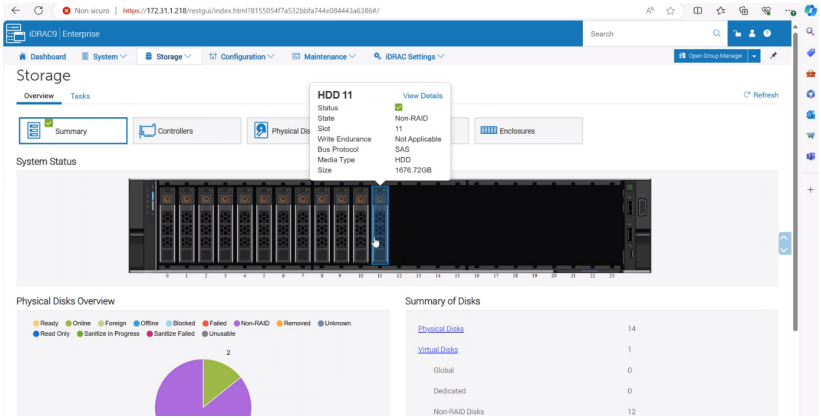

Figure 6.2: Monitoring storage
SAS bus protocol is used for storage devices.

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

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

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.

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 Random 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, and is much slower than virtualization.
TODO more on this
Virtualization is a strong way of isolating things.
There are two kind 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, 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.

Similar to the ones 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

VMware is the leader in virtualization, but lately they have been changing pricings and licensing, which has made some customers unhappy.
Broadcom is 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

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.
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 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.

8.2 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.
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.
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.
- ... TODO

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 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.
Layers above the control layer have no clue where workloads are running, they just know that they do so somewhere, it is completely up to the control 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 a Unified Manager software, which handles, by means of APIs, the tools associated to

- ◇ Compute System management
- ◇ Fabric management
- ◇ Storage System management

9.3.1 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

9.3.2 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.

9.3.3 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.

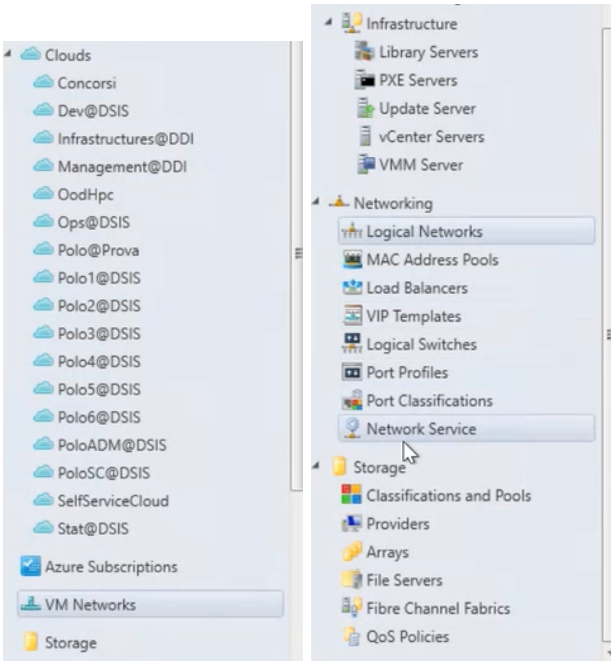

Figure 9.3: VM and Services view

He may even open a Powershell terminal on a VM.

This software allows him to manage about 1400 VMs.

9.4 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.

9.4.1 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.

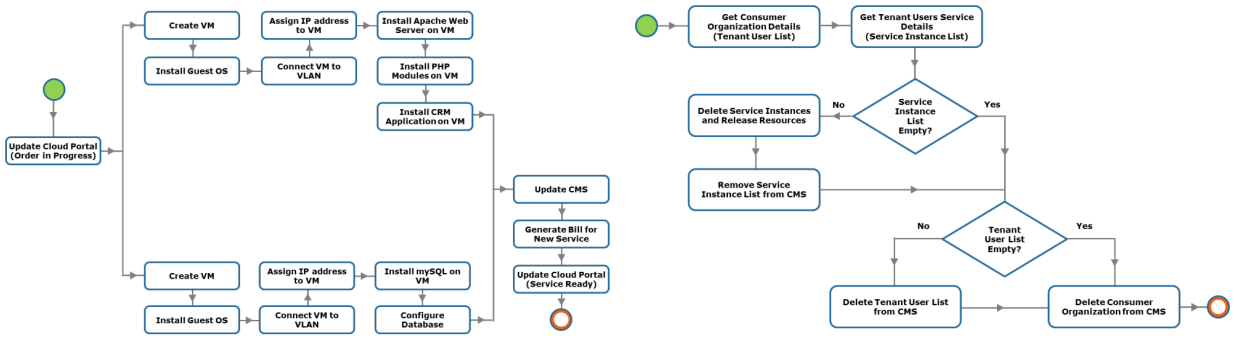

Figure 9.1: Service orchestration layer use cases. On the left, provisioning a CRM application, while on the right a tenant removal

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.

9.4.2 Deeper into Services

We may generalize the **lifecycle** of a service as it is depicted in Figure 9.2, 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

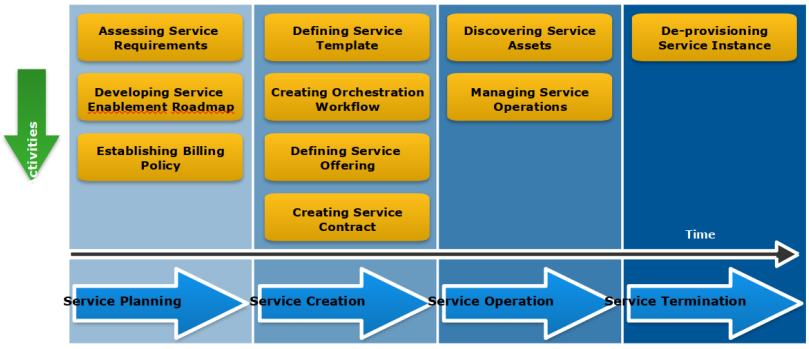

Figure 9.2: Service Lifecycle schema

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.

9.5 Business Continuity

Definition 9.4 (Business Continuity) *The capability of an organization to continue delivery of 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

SPOFs may occur at component level, or at site or 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 later).

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.
The implementation may be *active-active* or *active-passive*.

9.5.1 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.
"Backup must allow to travel back in time."

The two critical terms are *Recovery Time Objective* (`RTO`) and *Recovery Point Objective* (`RPO`)[1]. 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.

9.6 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.

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

[1] Point in time

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".

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.

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.

9.6.1 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.

9.6.2 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.

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.

Definition 9.6 (Zero Trust Architecture) *"Never trust, always verify"*

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

9.6.3 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.

9.6.4 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."

9.6.5 IDS and IPS

Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) are security measures that are used to protect a network from attacks.

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.

9.7 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.