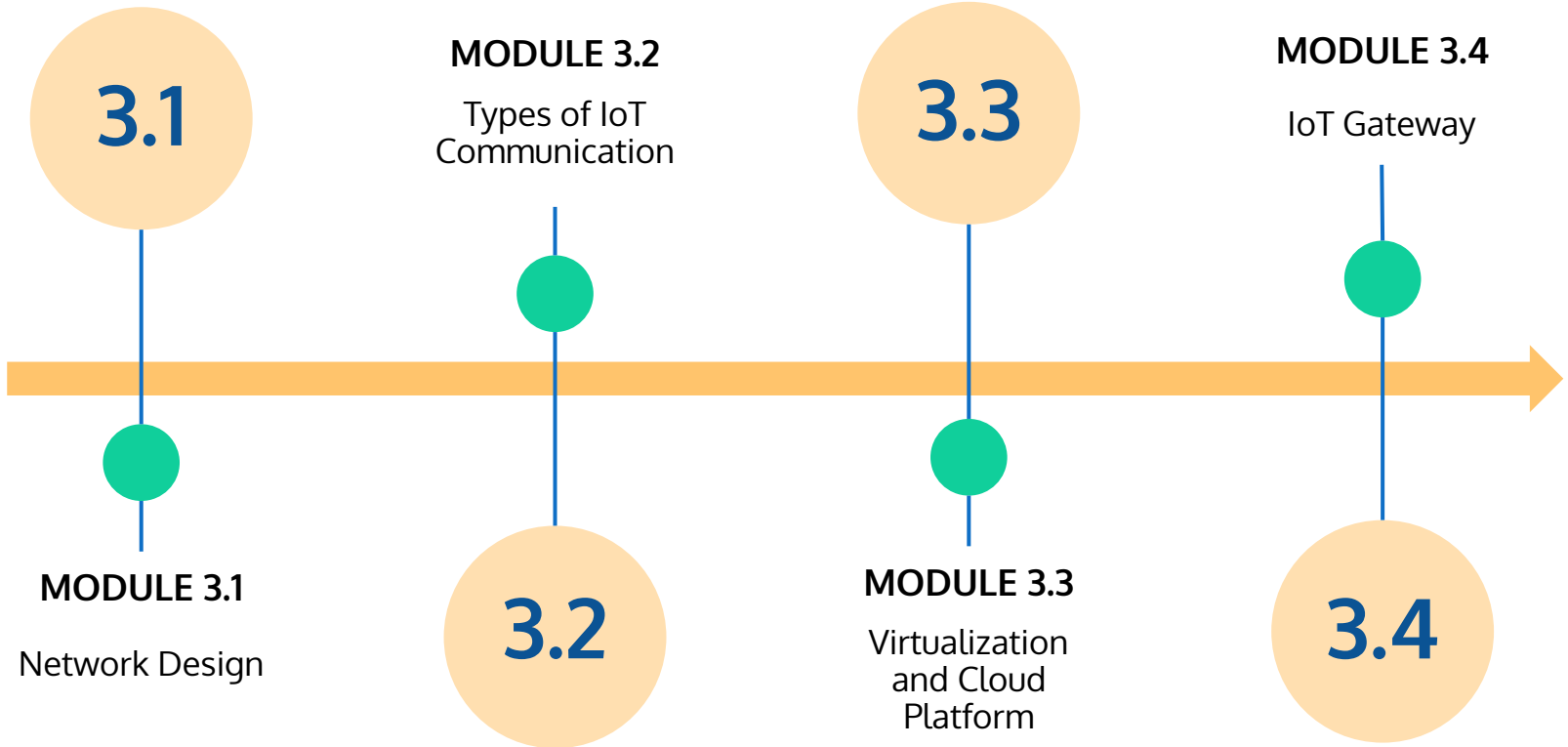
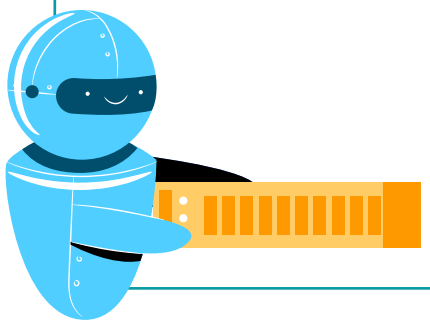
# Module 3 : IoT IT Infrastructure

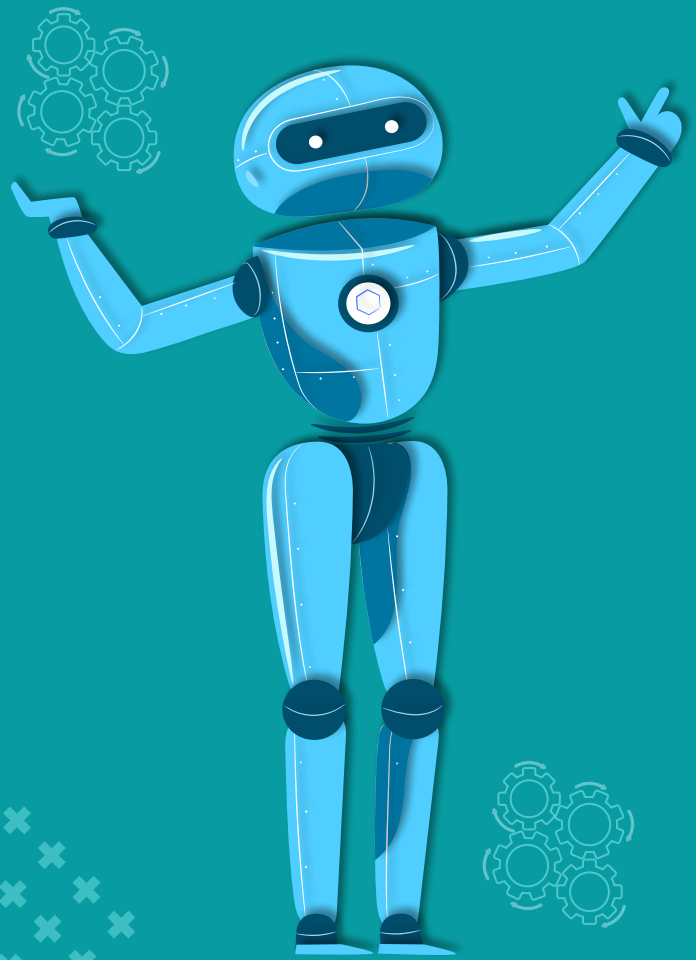
Presenter: Assoc. Prof. Ts Dr. Ahmad Shukri Mohd Noor

# MODULE OUTLINE



3.1



# Network Design

- National 2020 Budget - MITI
- Industry Automation Pyramid
- Underlying Technology
- Network Technology

# 3.1 Network Design

## 3.1.1 National 2020 budget - MITI

Malaysia is well positioned to be a primary destination for smart manufacturing and high technology activities...

MITI NST Business - May 7, 2019

"..nation needs to prepare well, and prepare fast, in order to be a winner in the digital economy.."

National 2020 Budget - Malaysia's Minister of Finance

- RM5.9 billion 2020 Nasional Budget - Technical and Vocational Education

# 3.1 Network Design

## 3.1.2 Industry Automation Pyramid


**Enterprise Resource Planning**

**Manufacturing Execution System**

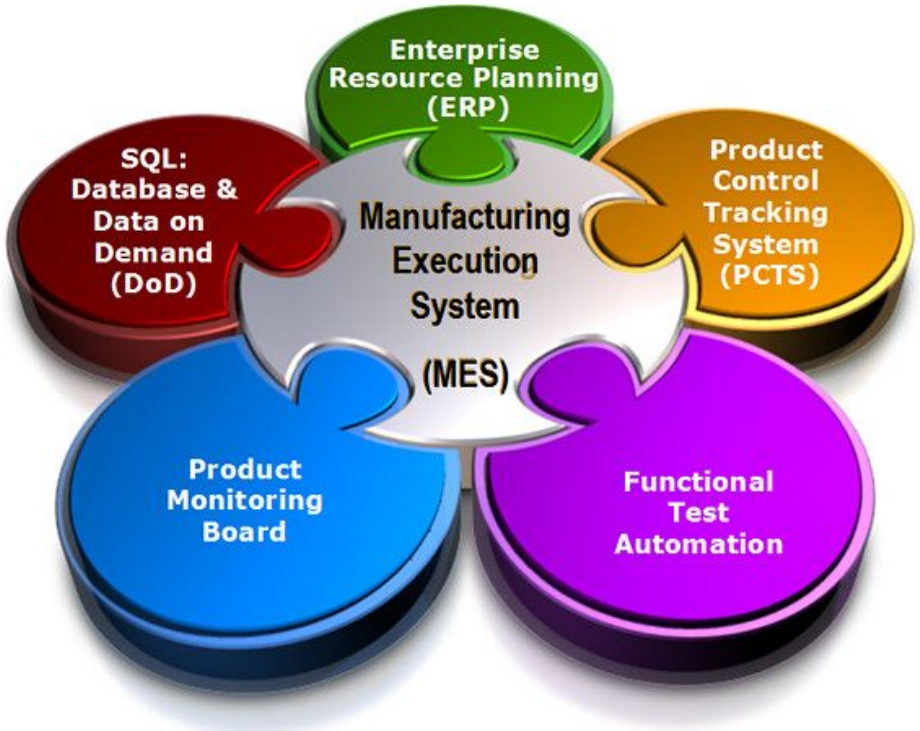
**SCADA & HMI**

**PLC and PID**

**actuator, sensors, motors, pneumatics, hydraulics, photo switches**

# 3.1 Network Design

## 3.1.2 Industry Automation Pyramid - MES




- Electronic and computer based system
- Executes Production Plan
- Enable data analysis for quality process such as Six Sigma and LEAN manufacturing.
- Capable of handling high-mixed-low volume manufacturing

# 3.1 Network Design

## 3.1.2 Industry Automation Pyramid - MES


- Financial Management
- SCM - Supply Chain Management
- Manufacturing Resource Planning
- Human Resource Mgt
- CRM
- Logistic
- Warehousing

# 3.1 Network Design

## 3.1.2 Industry Automation Pyramid - MES

Supply chain management is the management of the flow of goods and services and includes all processes that transform raw materials into final products.

A supply chain starts with the delivery of raw materials from a supplier to a manufacturer and ends with the delivery of the finished product or service to the end consumer.

# 3.1 Network Design

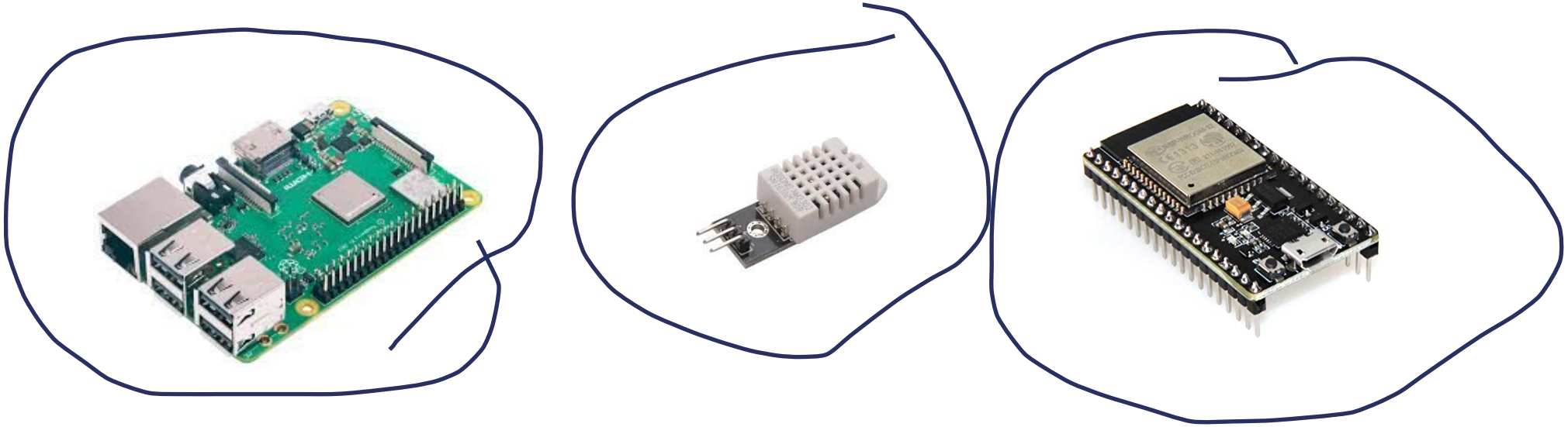
## 3.1.3 Underlying Technology

- E&E
- Electrical Power - VFD
- Mechatronics
- Micro electronics and Digital System
- Robotics and Motion
- Control system - SCADA and PLC
- Transducers and Sensors
- Actuators, relays and smart switches
- Embedded system and microcontrollers
- Cyber Physical System

- Information Technology Stacks for infrastructure and security.
- Data management technology SQL, NoSQL, Big DATA
- Cloud computing & Virtualization
- Data communication protocols for H2M, M2M, B2B
- Advanced Message Queuing Protocol
- Web App & Mobile development for UI/UX
- Back-End development for Business Process, eCommerce, etc
- Data analytics and AI.

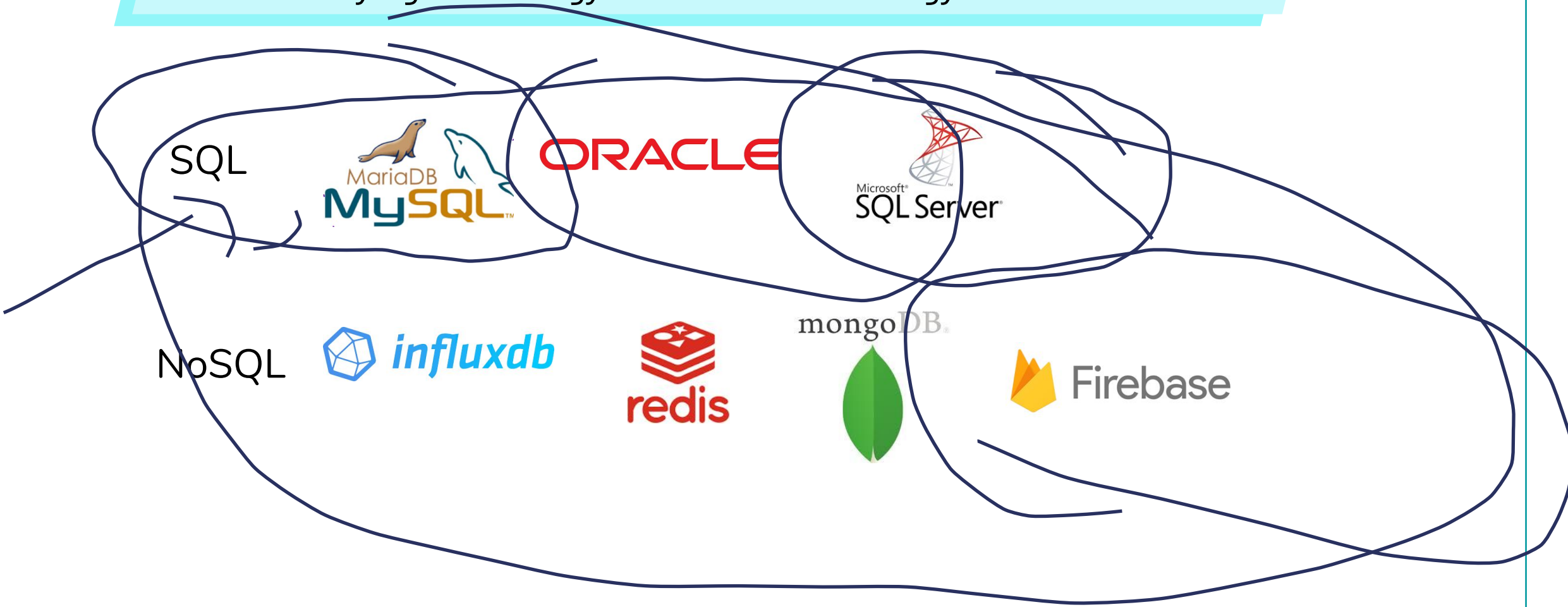
# 3.1 Network Design

## 3.1.3 Underlying Technology - IoT Cyber Physical System



# 3.1 Network Design

## 3.1.3 Underlying Technology - Database Technology



# 3.1 Network Design

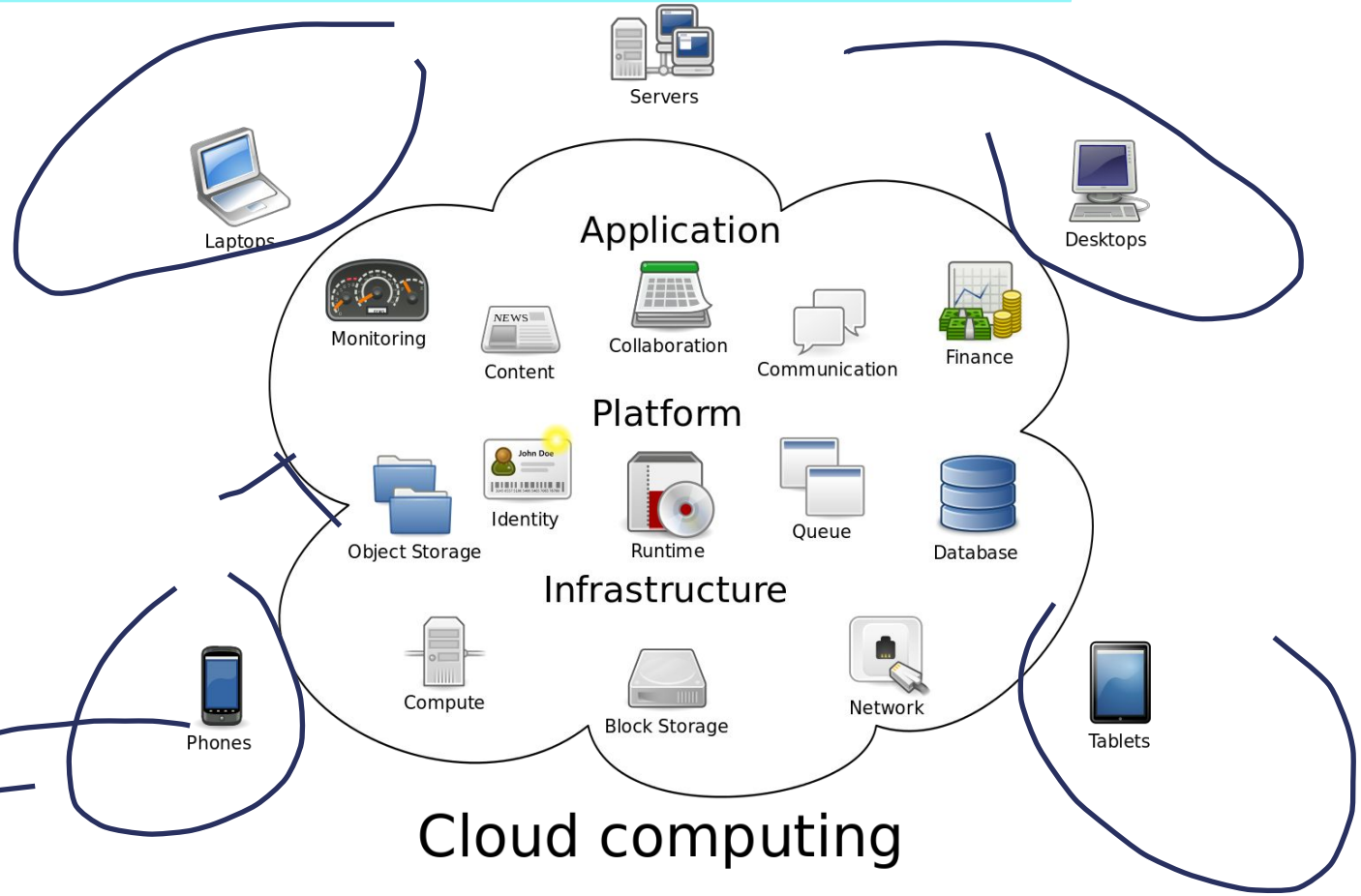
## 3.1.3 Underlying Technology - Cloud and Virtualization

Keys advantages:

1. To achieve seamless integration between business entities.
2. Anywhere - anytime - everyone
3. Machine to Machine
4. Human to Machine
5. Automated business process
6. Effective use of resources - human, machines, materials

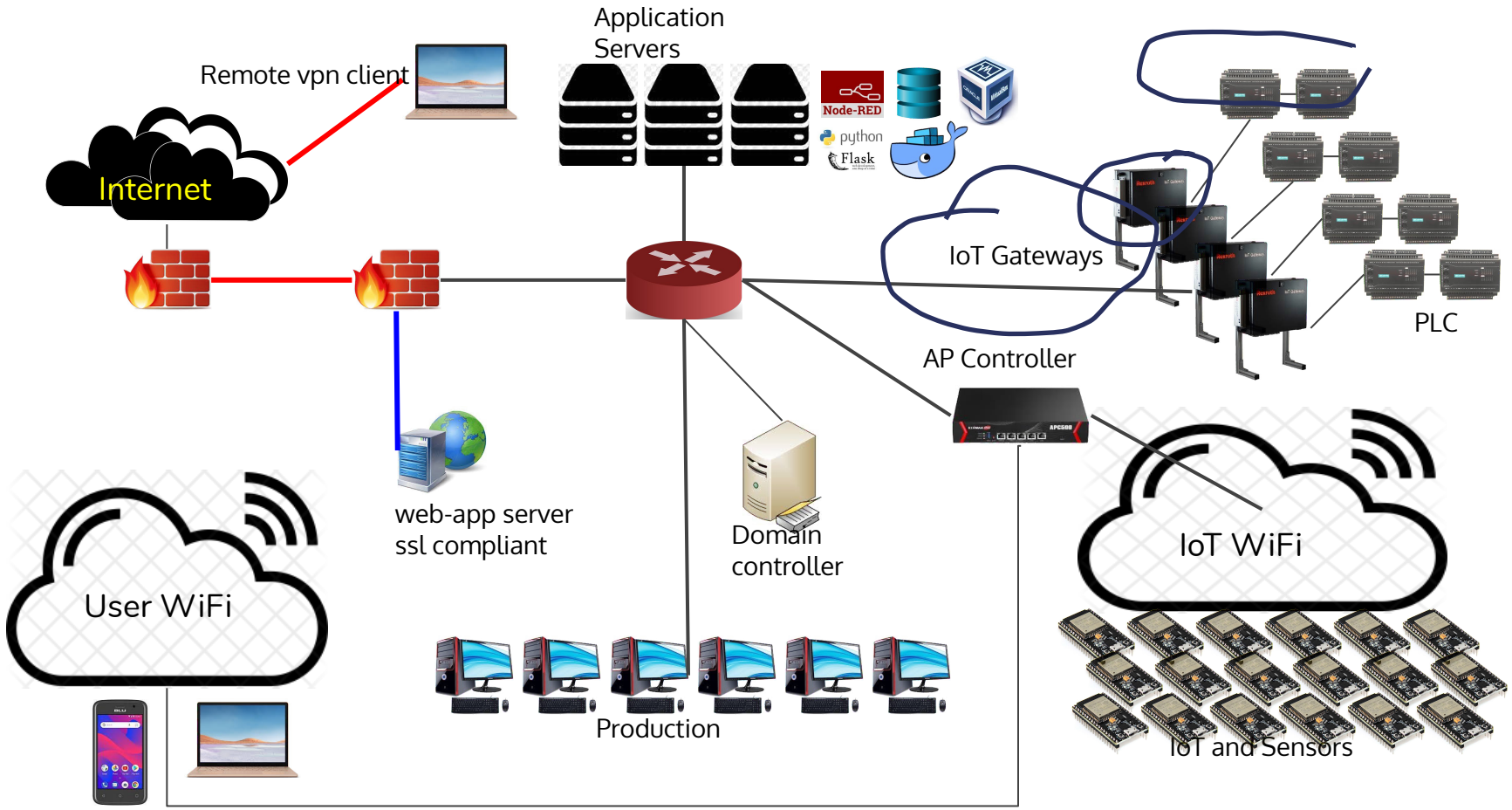
Drawbacks:

1. Securities
2. Lack of expertise
3. Partner readiness / resistance to change
4. Need investment



# 3.1 Network Design

## 3.1.3 Underlying Technology - Network Structure



# 3.1 Network Design

## 3.1.4 Network Technology

- Structured Cabling
- Ethernet and Network Switching Technology and WiFi
- OSI layer network model
- TCP / IP Addressing
- VLAN and VLAN Trunking technology
- Designing TCP/IP network (IP addressing)
- Inter VLAN routing
- Routing Protocol and Configuring Network Routing
- IT Services and TCP/IP Port number
- Port forwarding using firewall rules

# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling

- Structured cabling is the design and installation of a cabling system that will support multiple hardware uses to be suitable for today's needs and those of the future.

- With a correctly installed system, current and future requirements can be met, and hardware that is added in the future will be supported.

- Structured cabling design and installation is governed by a set of standards:
  - Specify wiring data centers, offices, and apartment buildings for data or voice communications using various kinds of cable, most commonly category 5e (Cat 5e), category 6 (Cat 6), and fiber optic cabling and modular connectors.
  - Define how to lay the cabling in various topologies in order to meet the needs of the customer, typically using a central patch panel (which is normally 19-inch rack-mounted), from where each modular connection can be used as needed. Each outlet is then patched into a network switch (normally also rack-mounted) for network use or into an IP or PBX (private branch exchange) telephone system patch panel.

*The Telecommunications Industry Association (USA) ANSI/TIA-568

# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling

Structured cabling consists of following 6 sub-systems:

1. Entrance Facilities – the point where the telephone company network ends and connects with the on-premises wiring at the customer premises
2. Equipment Rooms – host equipment which serves the users inside the building
3. Telecommunications Rooms – where various telecommunications and data equipment resides, connecting the backbone and horizontal cabling subsystems
4. Backbone cabling – inter and intra-building cable connections; it carries that signals between the entrance facilities, equipment rooms, and telecommunication rooms
5. Horizontal cabling – the wiring from the telecommunications rooms to the individual outlets on the floor
6. Work-Area components – connect end-user equipment to the outlets of the horizontal cabling system

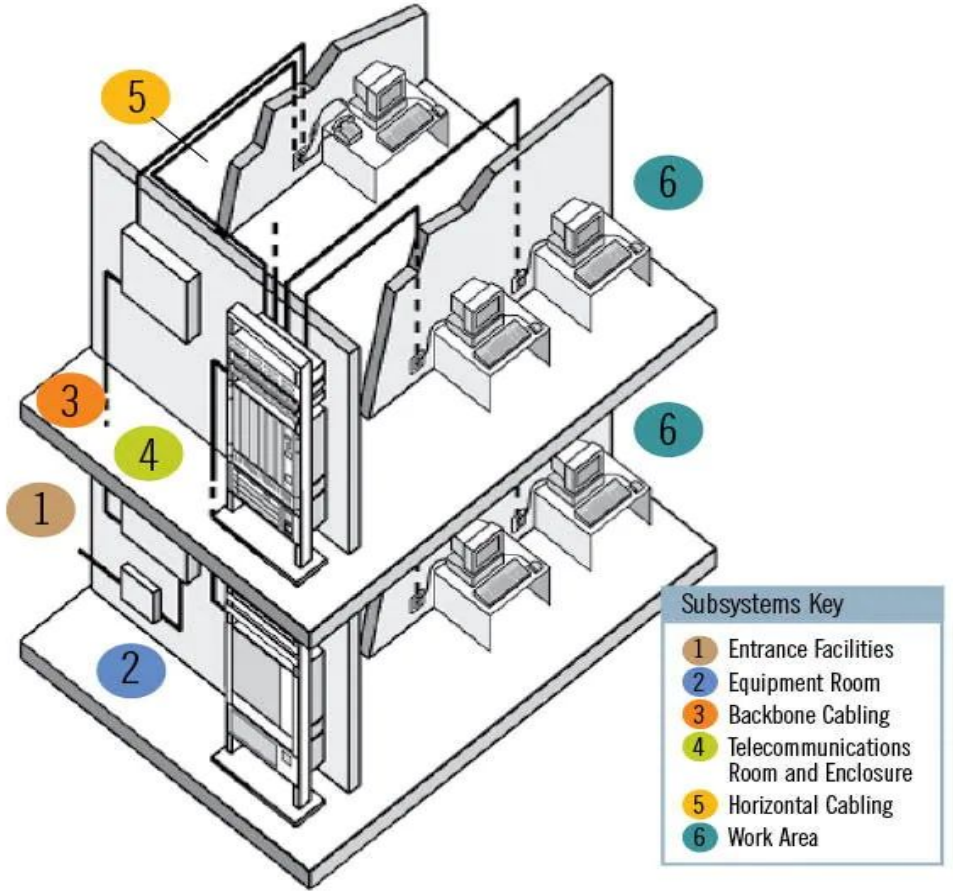
*The Telecommunications Industry Association (USA) ANSI/TIA-568

# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling

Structured cabling consists of following 6 sub-systems:

1. Entrance Facilities
2. Equipment Rooms
3. Telecommunications Rooms
4. Backbone cabling
5. Horizontal cabling
6. Work-Area components



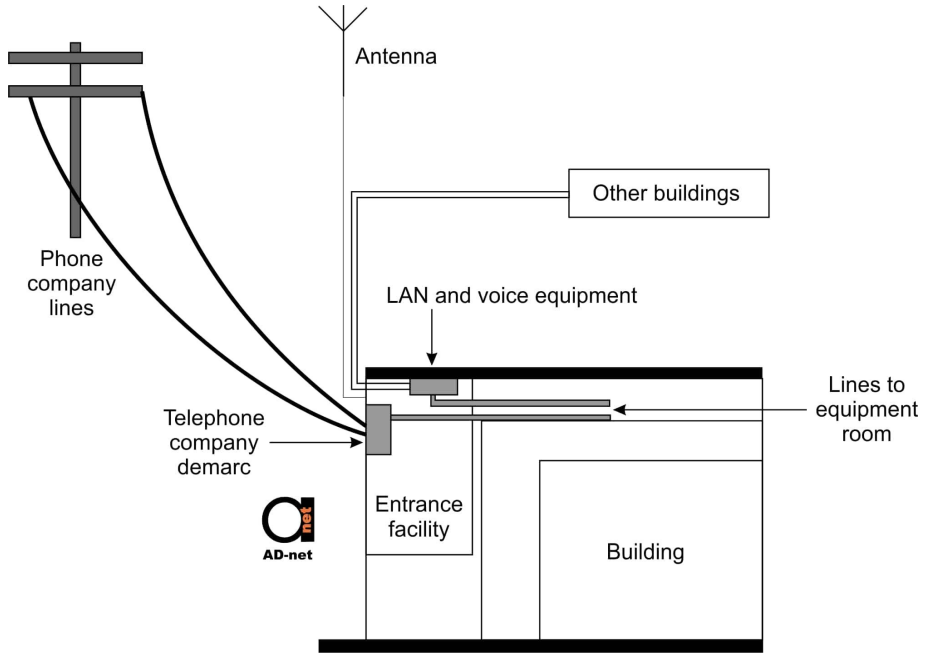
# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (entrance facility)

The entrance facility serves as the bridge between the Internet and your organization — it's the place where the connection between the local exchange carrier and your enterprise network begins.

What are needed:

1. Manhole / Pole  junction linking to the internet service provider (telco) system.
2. Grounding bus bar
3. Secure room
4. Enough Lighting
5. Trunking to the equipment room.
6. Demarcation line (separation area between telcos facilities and building owner's equipment.



# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (equipment room)

The equipment room (ER) houses tele-communications systems, such as PBXs, servers,routers, switches, and other core electronic components as well as the mechanical terminations.

Things found in ER:

1. server and equipment racks
2. grounding bus-bar
3. air-conditioner system
4. uninterruptible power supplies
5. main raceways or main trunking
6. Temperature and humidity monitoring
7. Automatic fire extinguisher - water mister / Co2
8. CCTV
9. automatic door access

# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (Telecommunication Room)

Telecommunication Closet - formerly known as the telecommunications closet.
The telecommunications room (TR) houses all the equipment associated with connecting the backbone
wiring to the horizontal wiring.

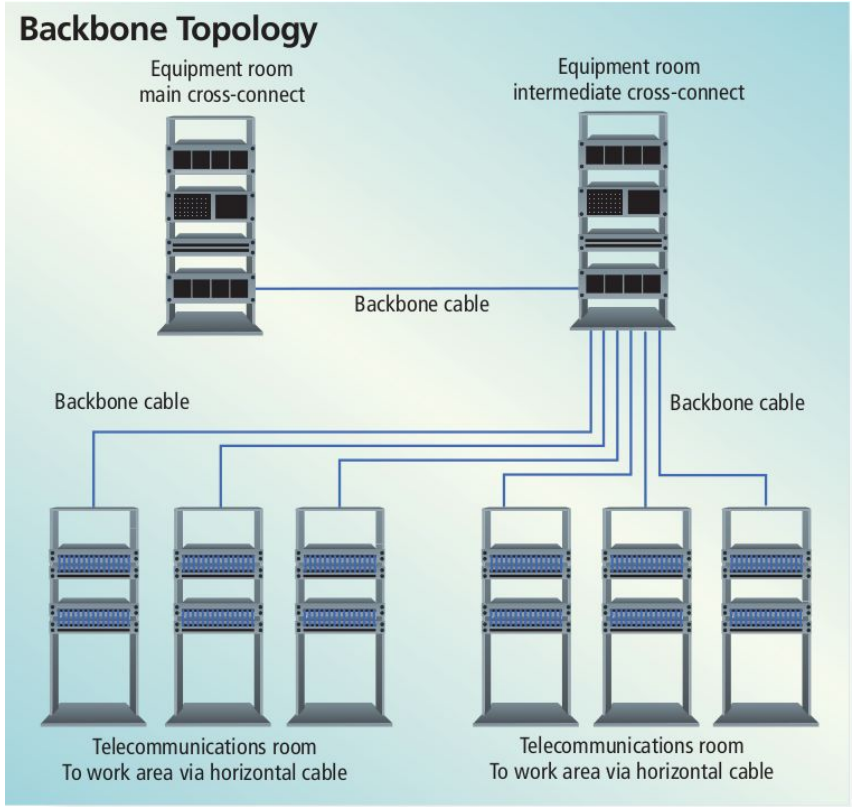
Things found in TR:

1. cabling rack / cabinet
2. patch panels, LIU
3. switches / routers
4. UTP, patch cords and fiber optics cables
5. power distribution panel

# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (Backbone Cabling)

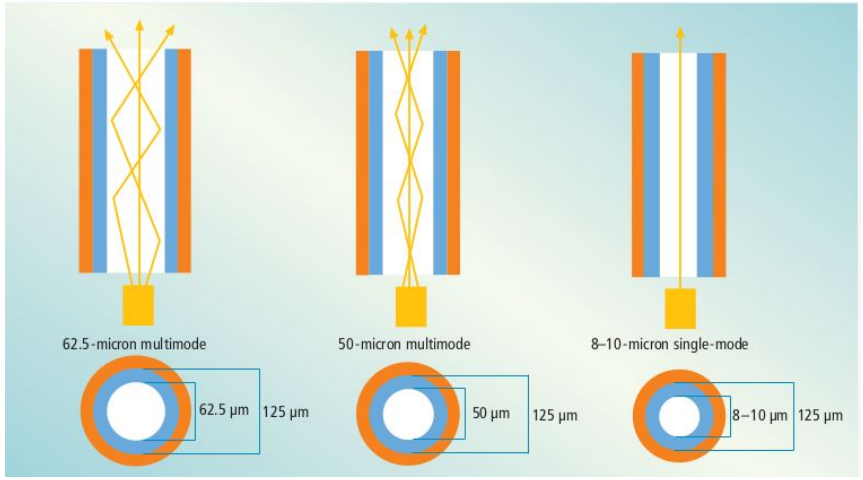
The backbone cabling system provides interconnections between telecommunications rooms, equipment rooms, main terminal space, and entrance facilities.
It includes backbone cables, intermediate and main cross-connects, mechanical terminations, and patch cords or jumpers used for backbone-to-backbone cross-connections.
The backbone also extends between buildings in a campus environment.

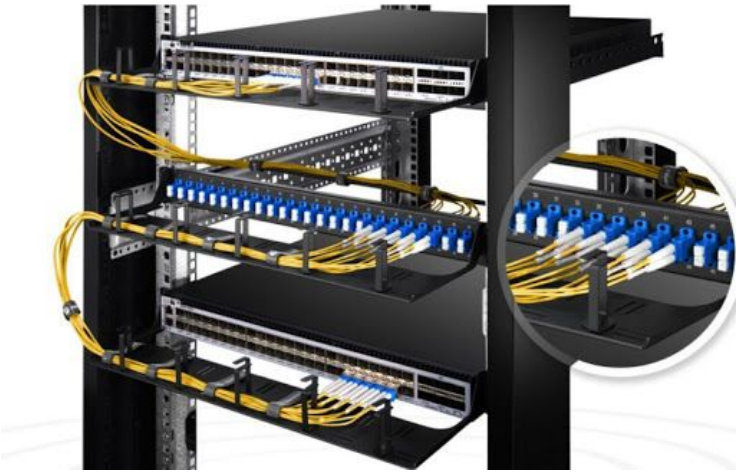


# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (Backbone Cabling)

| Backbone Cabling Distances | | | |
|---|---|---|---|
| Media Type | Main Cross-Connect to Horizontal Cross-Connect | Main Cross-Connect to Intermediate Cross-Connect | Intermediate Cross-Connect to Horizontal Cross-Connect |
| 100-ohm Copper | 800 m (2624.7 ft.) | 500 m (1640.4 ft.) | 300 m (984.3 ft.) |
| Multimode Fiber | 2000 m (6561.7 ft.) | 1700 m (5577.4 ft.) | 300 m (984.3 ft.) |
| Single-mode Fiber | 3000 m (9842.5 ft.) | 2700 m (8858.3 ft.) | 300 m (984.3 ft.) |

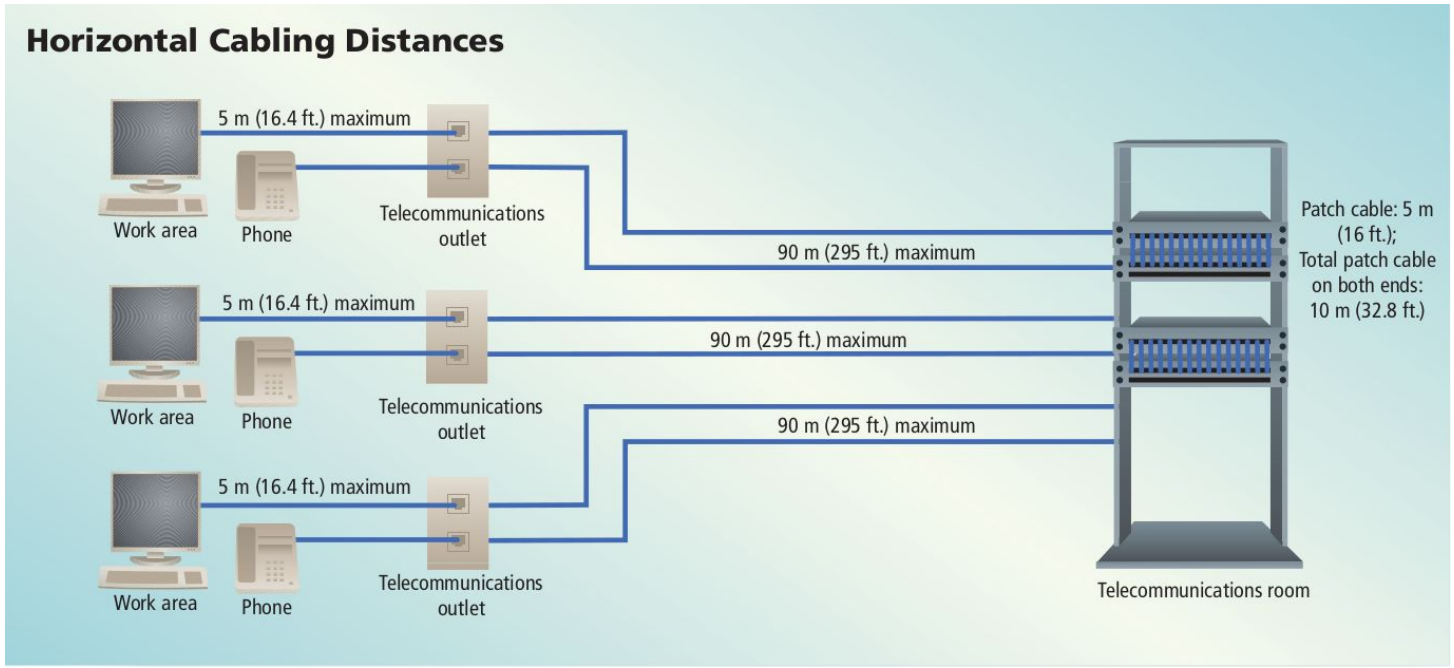


# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (Horizontal Cabling)

The horizontal cabling system refers to cabling between the telecommunications room cross-connects (patch panel & patch cords) to the telecommunications outlets in the work area. It's called horizontal because the cable typically runs horizontally above the ceiling or below the floor from the telecommunications room, which is usually on the same floor.



# 3.1 Network Design

## 3.1.4 Network Technology - Structured Cabling (Work Area Component)

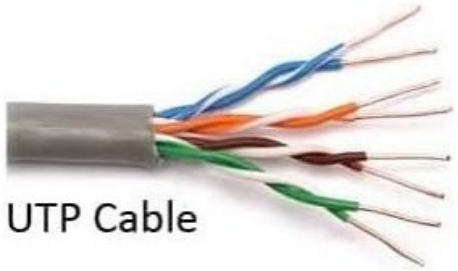
Work area (WA) components extend from the telecommunications outlet/connector end of the horizontal cabling system to the WA equipment.

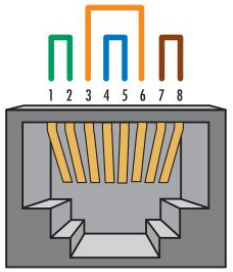
A minimum of two telecommunications outlets (permanent links) should be provided for each work area.

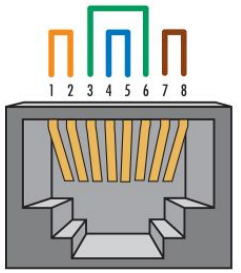
Your work desk or cubicle is example of place for work area components can be found.

# 3.1 Network Design

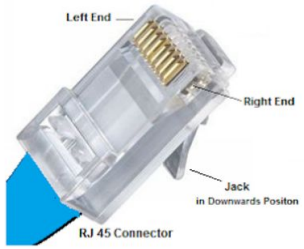
## 3.1.4 Network Technology - Structured Cabling (UTP, Connectors and Wiring Map)

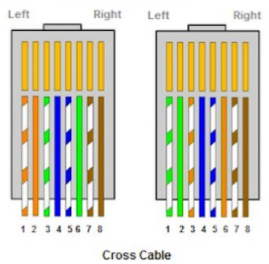







# 3.1 Network Design

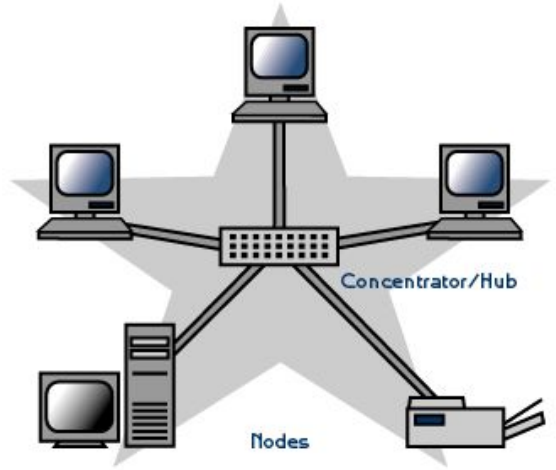
## 3.1.4 Network Technology - Ethernet and WiFi

Ethernet refers to the family of local-area network (LAN) products covered by the IEEE 802.3 standard that defines what is commonly known as the CSMA/CD protocol.
"Carrier-sense multiple access with collision detection (CSMA/CD) "

Currently there are four data speed supported by ethernet:

- 10 Mbps - 10Base-T Ethernet
- 100 Mbps - Fast Ethernet
- 1,000 Mbps - Gigabit Ethernet
- 10,000 Mbps - 10 Gigabit Ethernet

Ethernet uses media cables:
– Coaxial cable (which is not very common except in older installations)
– Twisted pair (category 3, 5, 6)
– Fiber optics




Implements Star topology:

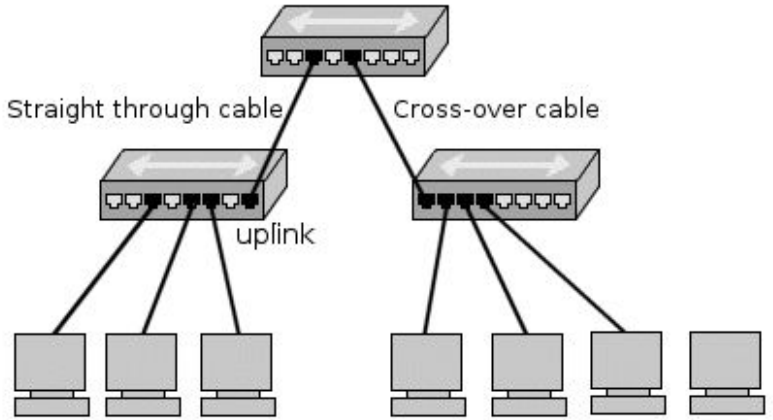
- Stable
- Scalable
- Deployed on structure cabling infrastructure

# 3.1 Network Design

## 3.1.4 Network Technology - Ethernet and WiFi

Ethernet Switches:
A network switch is networking hardware that connects devices on a computer network by using packet switching to receive and forward data to the destination device.

An Ethernet switch operates at the data link layer (layer 2) of the OSI model to create a separate collision domain for each switch port.
Each device connected to a switch port can transfer data to any of the other ports at any time and the transmissions will not interfere.
Ethernet switches can be connected in cascading manner to enable network expansions following start topology.
Structured cabling standards ensures that the switches can be deployed correctly and perform as designed.



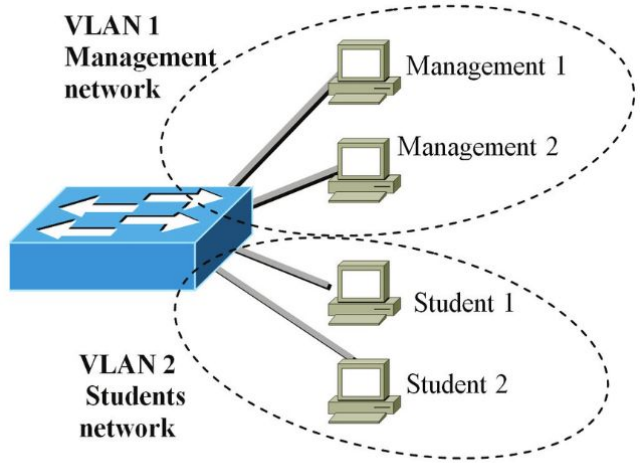
# 3.1 Network Design

## 3.1.4 Network Technology - Ethernet and WiFi

**Unmanaged** switches are typically used to provide basic connectivity. They're designed to be plug and play, no configuration is needed.

Unmanaged switches are most effective when only basic switching and connectivity are required. You will often see them in home networks or wherever only a few ports are needed, such as at a desk, in a lab, or in a conference room.

**Managed** switches are designed to deliver the most comprehensive set of features to provide high levels of security, precise control and management of the network, and scalability. Managed switches are usually deployed as **aggregation/access switches** in very **large networks** or as **core switches** in smaller networks. It supports **VLAN** assignment on every available port which user can configure as he likes according to LAN design and security requirements.




Two VLAN networks in a single switch.

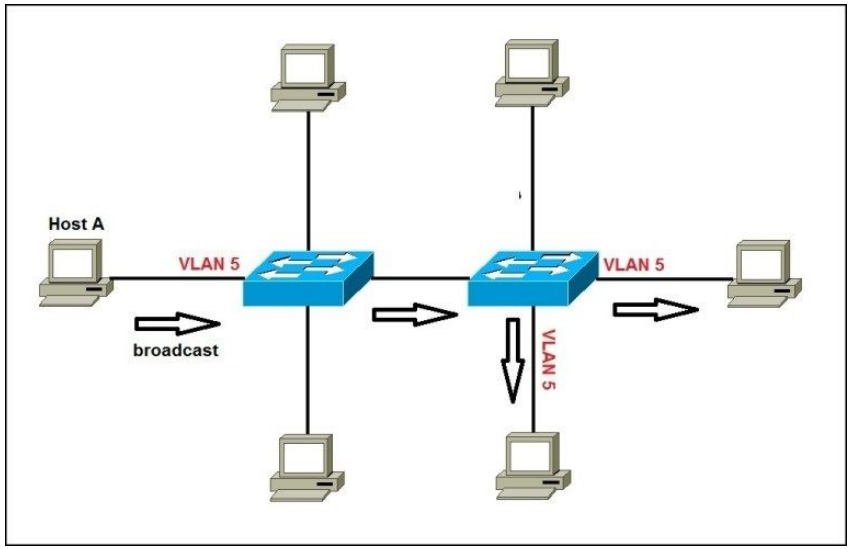
Managed switch controls LAN access using VLAN configuration

# 3.1 Network Design

## 3.1.4 Network Technology - Ethernet and WiFi (VLAN)

**VLANs** (Virtual LANs) are **logical grouping** of devices in the same broadcast domain. VLANs are usually configured on switches by placing some interfaces into one broadcast domain and some interfaces into another.
Each VLAN acts as a subgroup of the switch ports in an Ethernet LAN.

VLANs can **spread across multiple switches,** with each VLAN being treated as its own subnet or broadcast domain. This means that **frames broadcasted onto the network will be switched only between the ports within the same VLAN.**
A VLAN acts like a physical LAN, but it ***allows hosts to be grouped together in the same broadcast domain*** even if they are not connected to the same switch.




Example broadcast are DHCP and ARP requests

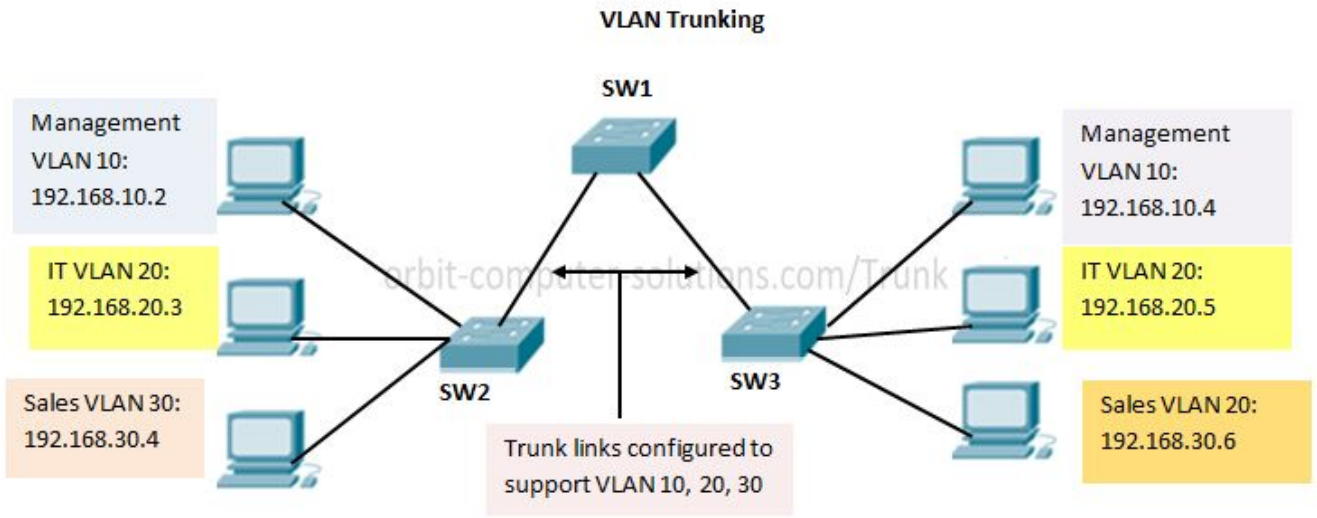
# 3.1 Network Design

## 3.1.4 Network Technology - Ethernet and WiFi (VLAN)

A VLAN trunk is a point-to-point link between two network devices that carry more than one VLAN.

With VLAN trunking, you can extend your configured VLAN across the entire network.

Most Cisco switches support the IEEE 802.1Q used to coordinate trunks on FastEthernet and GigabitEthernet.

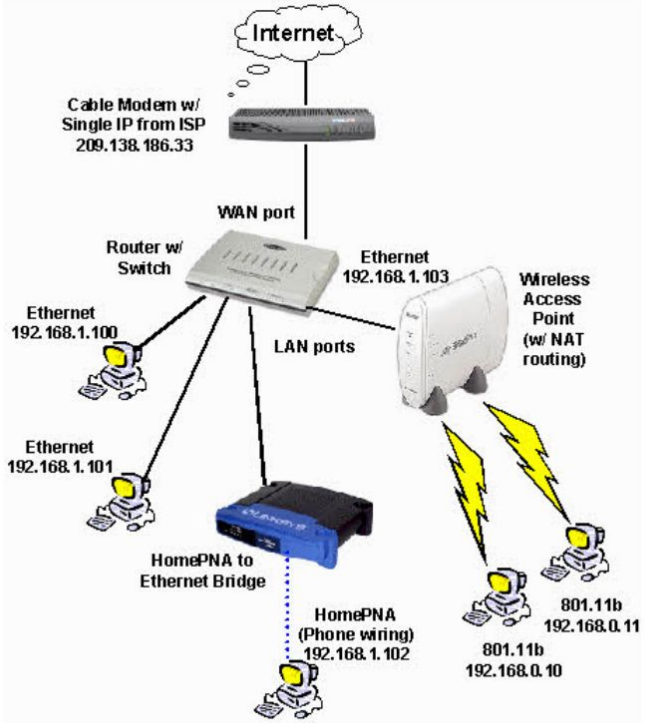


# 3.1 Network Design

## 3.1.4 Network Technology - Ethernet and WiFi (WLAN)

"wireless local area network (WLAN) products that are based on the Institute of Electrical and Electronics Engineers' (IEEE) 802.11 standards."

| Type | Speed |
|---|---|
| 802.11a | 54 Mbps is the maximum, but usually 6 to 24 Mbps |
| 802.11b | 11 Mbps |
| 802.11g | 54 Mbps |
| Wireless-N | 72-300 Mbps |
| Wireless-AC | 433-1733 Mbps |
| Wireless-AX | 600-2402 Mbps |




*Wireless Network*

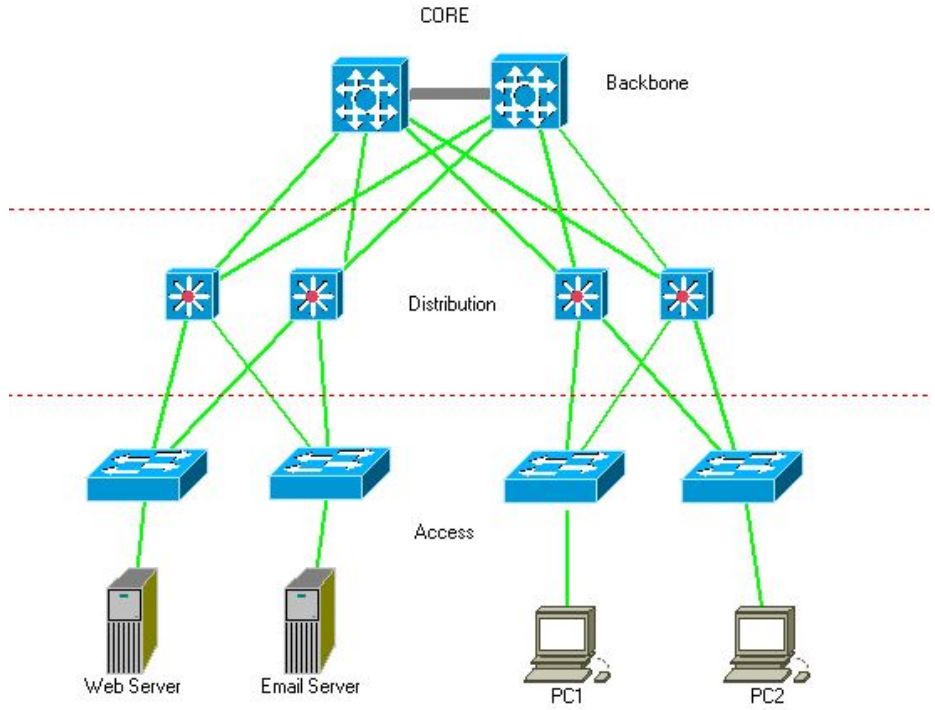
# 3.1 Network Design

## 3.1.4 Network Technology - Hierarchical internetworking model (Ethernet Core, Distribution, Access)

The **core network** provides high-speed, highly redundant forwarding services to move packets between distribution-layer devices in different regions of the network. The switches are the most powerful in forwarding network packets to handle network traffic from access and distribution layers via its core layer. The hardware is fully fault tolerant usually having redundant power supplies and supervisor engines.

The **distribution layer** is the smart layer in the three-layer model. Routing, filtering, and QoS policies are managed at the distribution layer. Distribution layer devices also often manage individual branch-office WAN connections. This layer is also called the Workgroup layer.

End-stations such as users desktop and servers connect to the enterprise at the **access layer**. Access layer switch are usually layer 2 switches that implements VLAN on each port to isolate networks and increase performance by reducing broadcasts and collisions, hence adding most basic security in network design.

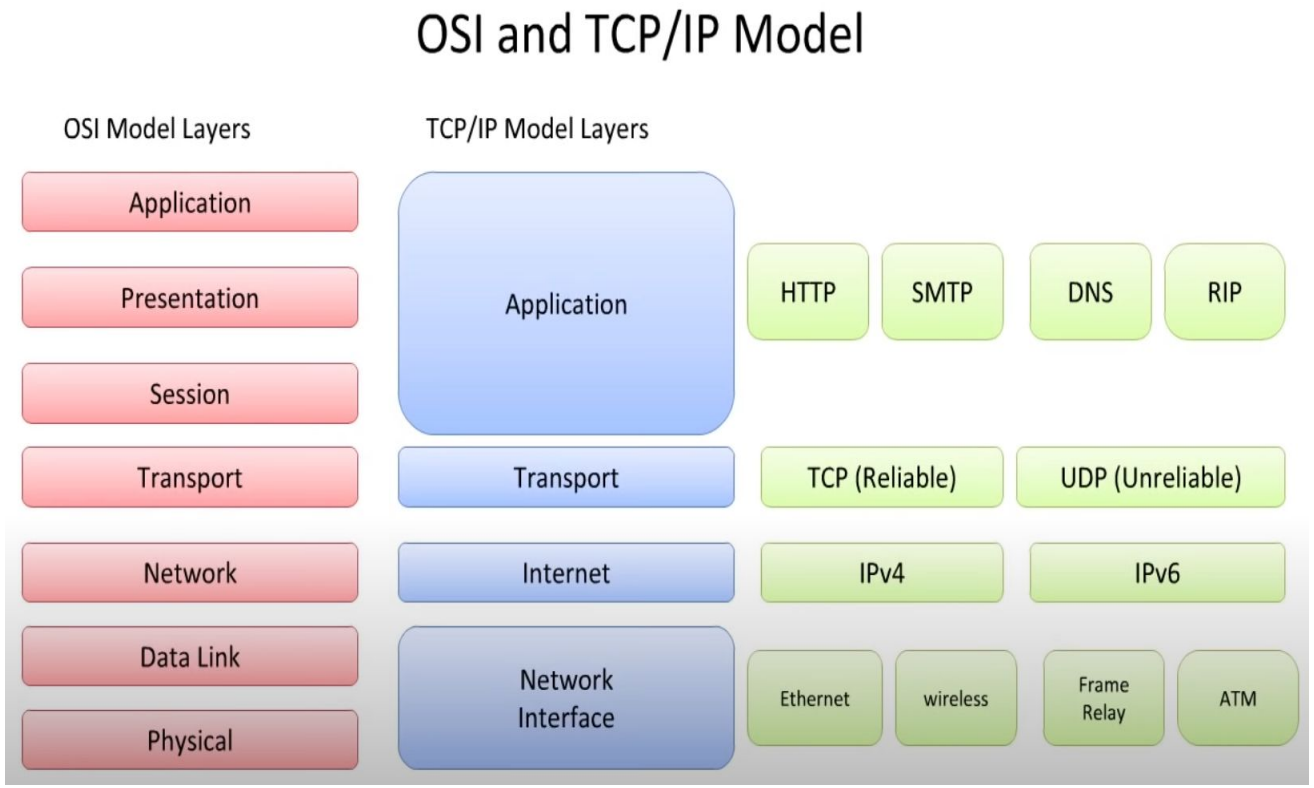


# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

The Open Systems Interconnection model (OSI model) is a conceptual model that characterises and standardizes the communication functions of a telecommunication or computing system without regard to its underlying internal structure and technology.

Its goal is the interoperability of diverse communication systems with standard communication protocols.

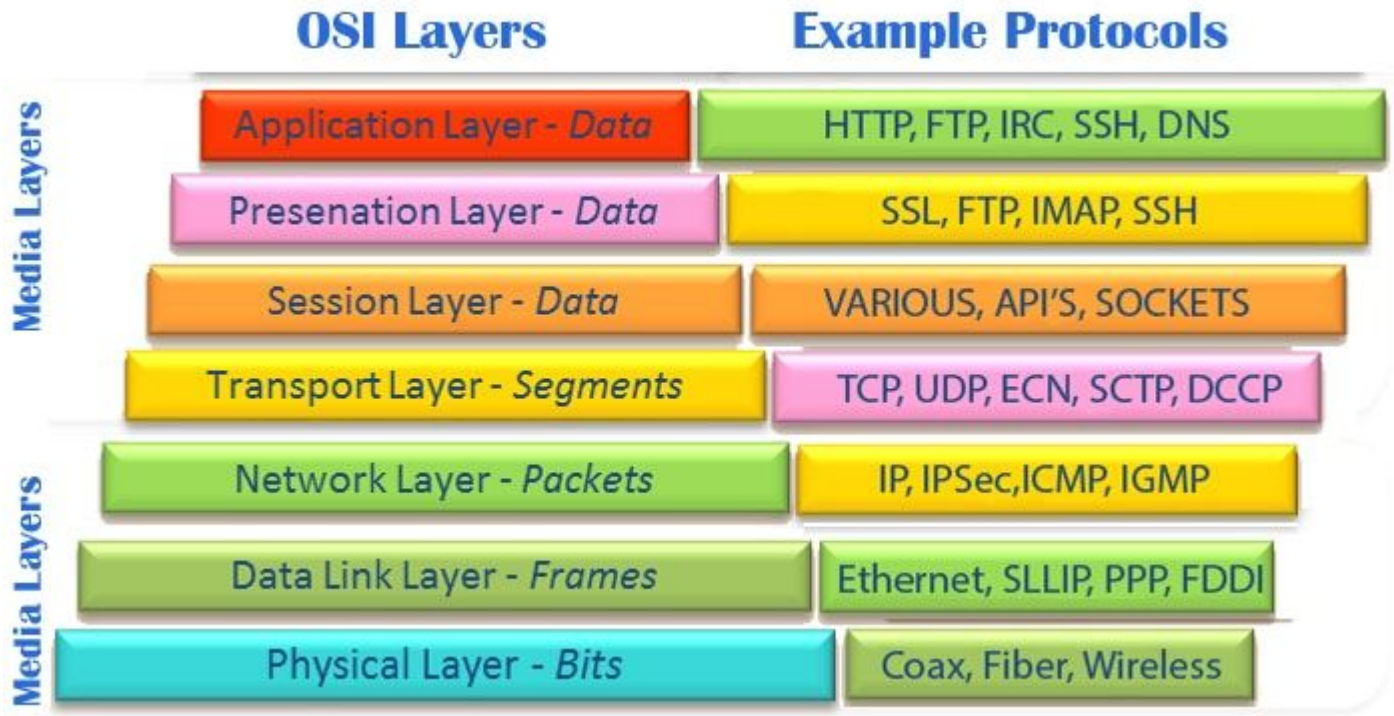


# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

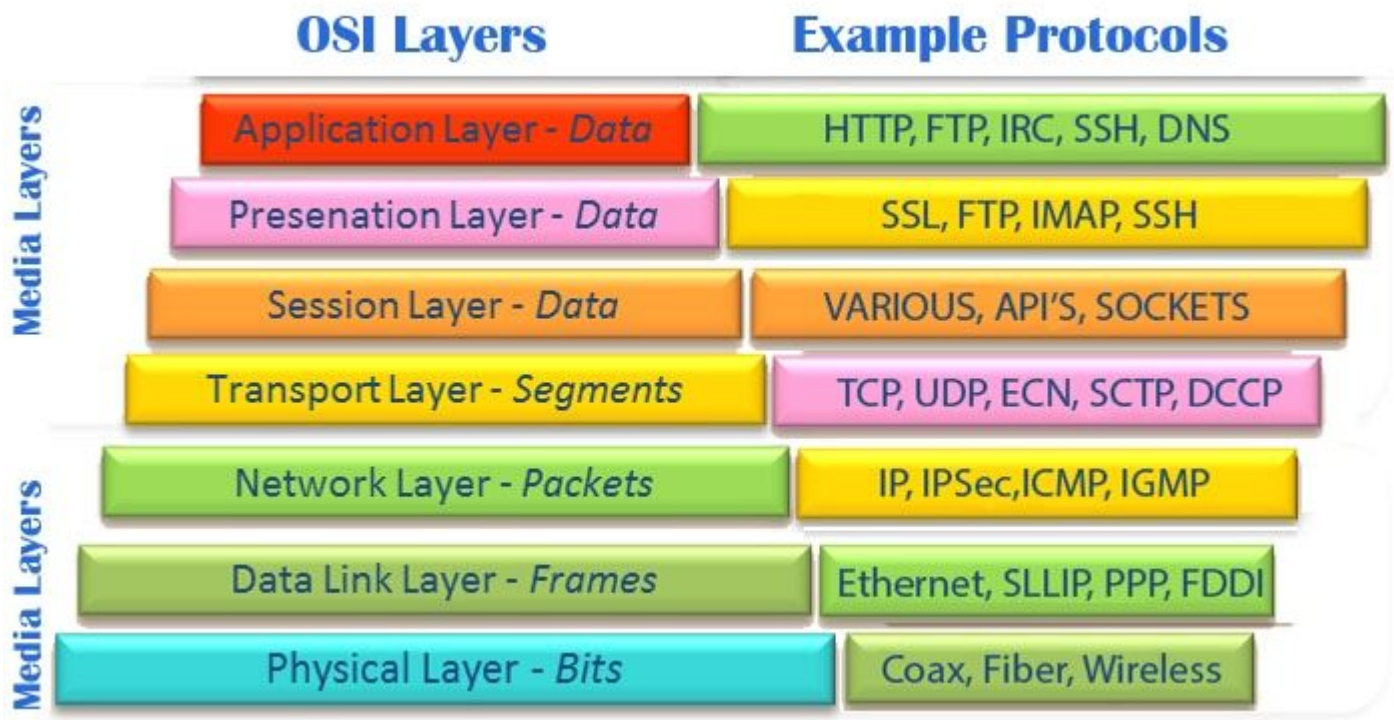
Layer 7 - Application
The Application Layer is the one at the top - it's what most users see. In the OSI model, this is the layer that is the "closest to the end user". Applications that work at Layer 7 are the ones that users interact with directly.

Layer 6 - Presentation
The Presentation Layer represents the area that is independent of data representation at the application layer - in general, it represents the preparation or translation of application format to network format, or from network formatting to application format. In other words, the layer "presents" data for the application or the network.



## 3.1.4 Network Technology - OSI Layer Network Model

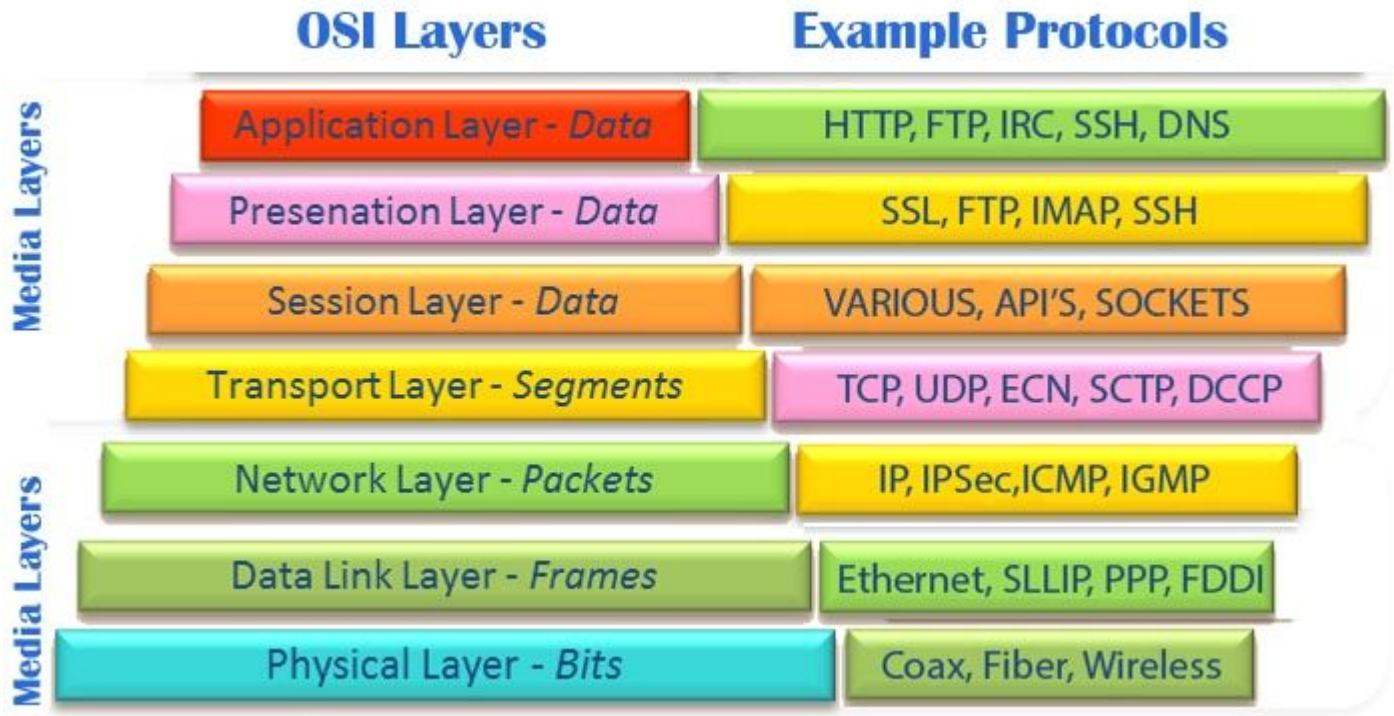
Layer 5 - Session
When two devices, computers or servers need to "speak" with one another, a session needs to be created, and this is done at the Session Layer. Functions at this layer involve setup, coordination (how long should a system wait for a response, for example) and termination between the applications at each end of the session.



# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

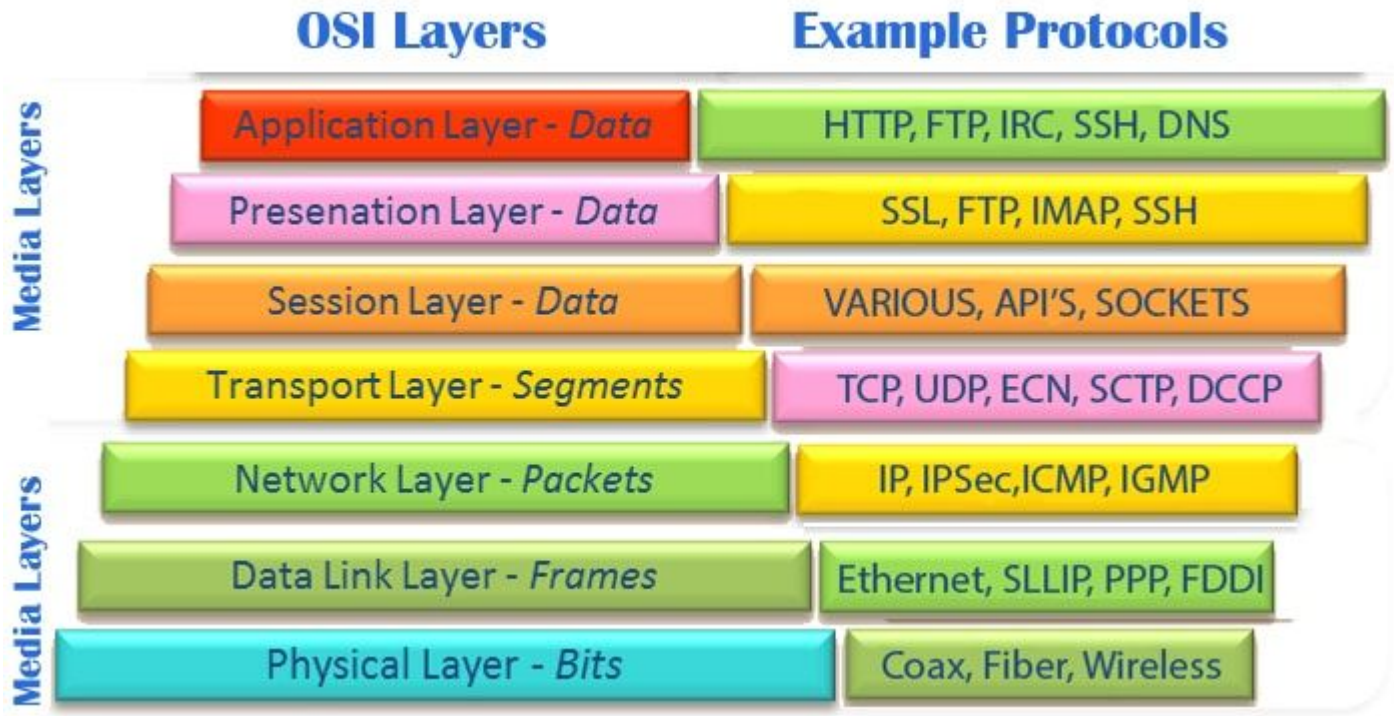
Layer 4 – Transport
The Transport Layer deals with the coordination of the data transfer between end systems and hosts. How much data to send, at what rate, where it goes, etc. The best known example of the Transport Layer is the Transmission Control Protocol (TCP), which is built on top of the Internet Protocol (IP), commonly known as TCP/IP. TCP and UDP port numbers work at Layer 4, while IP addresses work at Layer 3, the Network Layer.



# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

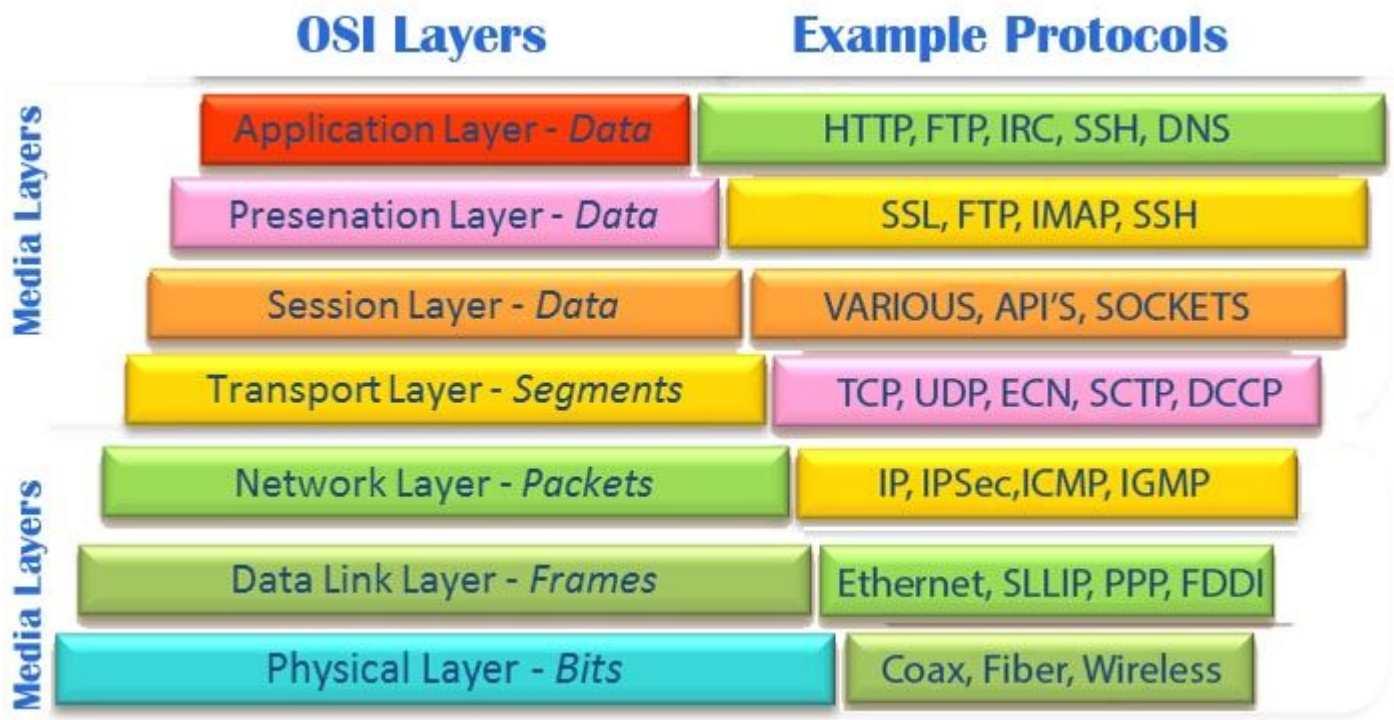
Layer 3 - Network
Here at the Network Layer is where you'll find most of the router functionality that most networking professionals care about. In its most basic sense, this layer is responsible for packet forwarding, including routing through different routers. Data packet need to find its destination address via available network connections or nodes. Routers at this layer help identify the connection path efficiently.



# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

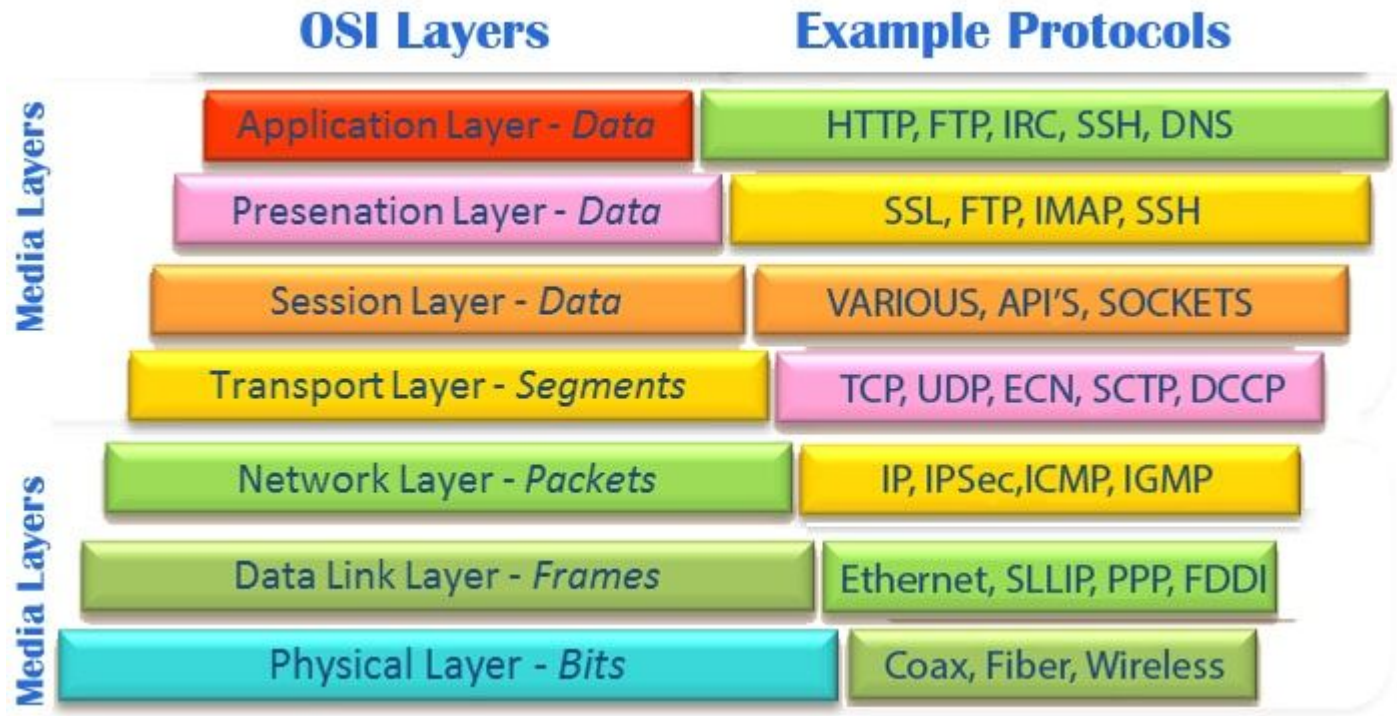
Layer 2 – Data Link
The Data Link Layer provides node-to-node data transfer (between two directly connected nodes), and also handles error correction from the physical layer. Two sublayers exist here as well - the Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. In the networking world, most switches operate at Layer 2.



# 3.1 Network Design

## 3.1.4 Network Technology - OSI Layer Network Model

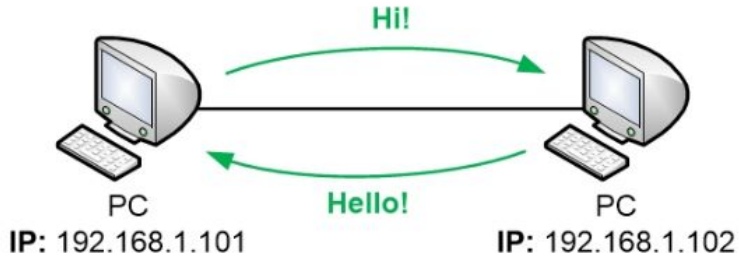
Layer 1 - Physical
At the bottom of our OSI bean dip we have the Physical Layer, which represents the electrical and physical representation of the system. This can include everything from the cable type, radio frequency link (as in an 802.11 wireless systems), as well as the layout of pins, voltages and other physical requirements. When a networking problem occurs, many networking pros go right to the physical layer to check that all of the cables are properly connected and that the power plug hasn't been pulled from the router, switch or computer, for example.



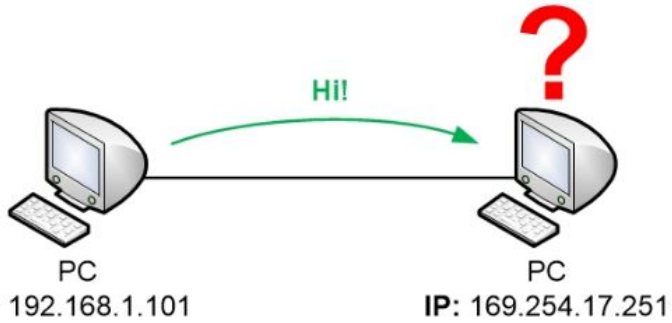
# 3.1 Network Design

## 3.1.4 Network Technology - IP Addressing

TCP/IP protocol uses IP address to identify network devices connected to the network.
The address must be assigned uniquely to each host and it is a temporary in nature.
A switch remembers the IP addresses and map to the built in hardware address (Media Access Control address).
Address Resolve Protocol helps switches to discover the connected hosts MAC and IP addresses and reuse them for future data communication.



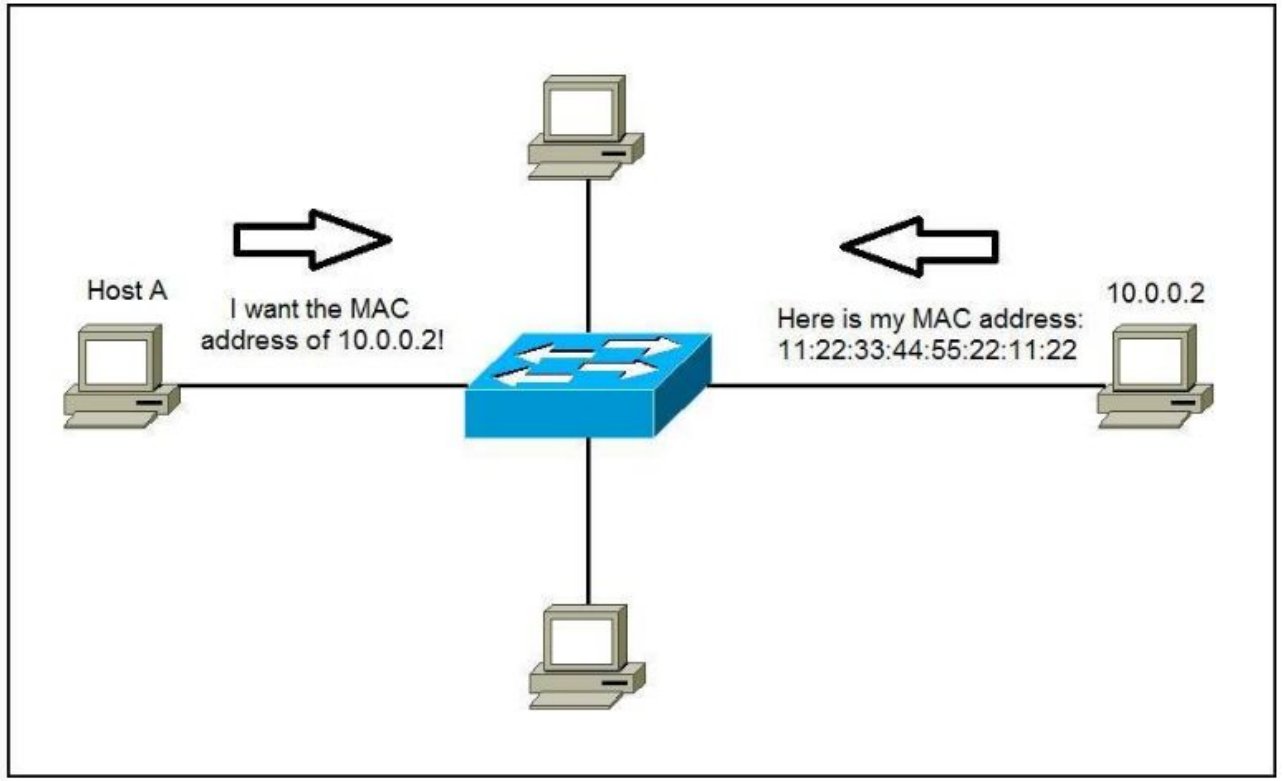

*Two computers belonging to the same IP network*




*Two computers belonging to different IP networks*

# 3.1 Network Design

## 3.1.4 Network Technology - IP Addressing ARP



# 3.1 Network Design

## 3.1.4 Network Technology - IP Addressing and Subnetting

An IP address is a 32-bit number that uniquely identifies a host (computer or other device, such as a printer or router) on a TCP/IP network.
IP addresses are normally expressed in dotted-decimal format, with four numbers separated by periods, such as 192.168.123.132.
Each number also known as an Octet, represents one byte binary values from 0 to 255 decimal.

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

The binary number 1111 1111 converts into the decimal number:

**128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255**

# 3.1 Network Design

## 3.1.4 Network Technology - Calculating IP Address

An IP address is a 32-bit number that uniquely identifies a host (computer or other device, such as a printer or router) on a TCP/IP network.
IP addresses are normally expressed in dotted-decimal format, with four numbers separated by periods, such as 192.168.123.132.
Each number also known as an Octet, represents one byte binary values from 0 to 255 decimal.

| No. of Host Bits | Equivalent prefix length | Subnet Mask | Number of usable IP addresses |
|---|---|---|---|
| 1 | /31 | 255.255.255.254 | $2^1$-2 = 0* |
| 2 | /30 | 255.255.255.252 | $2^2$-2 = 2 |
| 3 | /29 | 255.255.255.248 | $2^3$-2 = 6 |
| 4 | /28 | 255.255.255.240 | $2^4$-2 = 14 |
| 5 | /27 | 255.255.255.224 | $2^5$-2 = 30 |
| 6 | /26 | 255.255.255.192 | $2^6$-2 = 62 |
| 7 | /25 | 255.255.255.128 | $2^7$-2 = 126 |
| 8 | /24 | 255.255.255.0 | $2^8$-2 = 254 |
| 9 | /23 | 255.255.254.0 | $2^9$-2 = 510 |
| 10 | /22 | 255.255.252.0 | $2^{10}$-2 = 1022 |

## 3.1.4 Network Technology - IP Address Type and CIDR

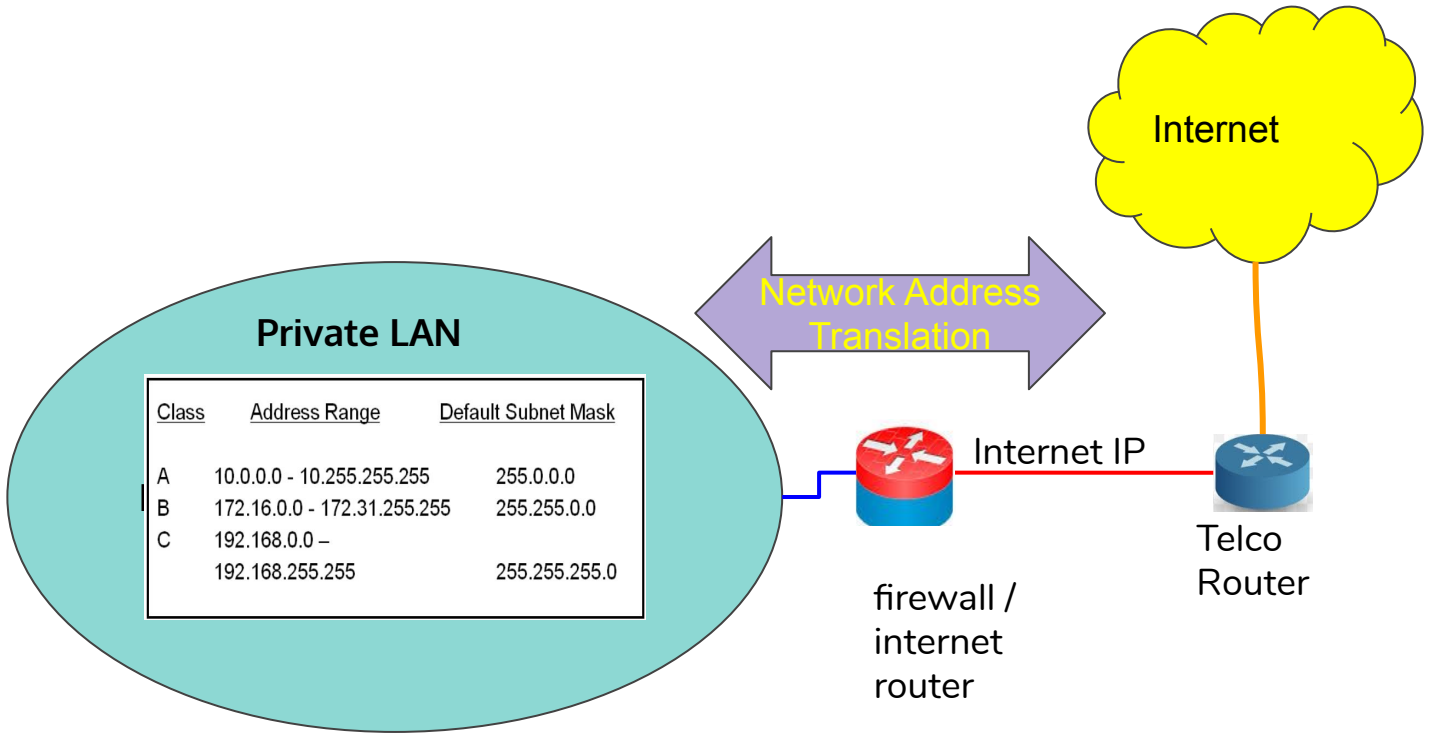
CIDR (Classless Inter-Domain Routing, sometimes known as supernetting) is a way to allocate and specify the Internet addresses used in inter-domain routing more flexibly than with the original system of Internet Protocol (IP) address classes. As a result, the number of available Internet addresses has been greatly increased. CIDR is now the routing system used by virtually all gateway hosts on the Internet's backbone network. The Internet's regulating authorities now expect every Internet service provider (ISP) to use it for routing.

| Class | From | To | CIDR Mask | Decimal Mask |
|---|---|---|---|---|
| Class "A" or 24 Bit | 10.0.0.0 | 10.255.255.255 | /8 | 255.0.0.0 |
| Class "B" or 20 Bit | 172.16.0.0 | 172.31.255.255 | /12 (or more typically /16) | 255.240.0.0 (or 255.255.0.0) |
| Class "C" or 16 Bit | 192.168.0.0 | 192.168.255.255 | /16 (or more typically /24) | 255.255.0.0 (or 255.255.255.0) |

# 3.1 Network Design

## 3.1.4 Network Technology - IP Address (Private IP vs Public IP addresses)

Telco / Internet routers will drop network packets that has private network address destinations.
We need to use firewall router to do network address translation (NAT) so that our private network devices can communicate with internet.
NAT is a service  performing dynamic mapping of allowed internal ip addresses to a single valid IP address assigned by telco to our internet router.



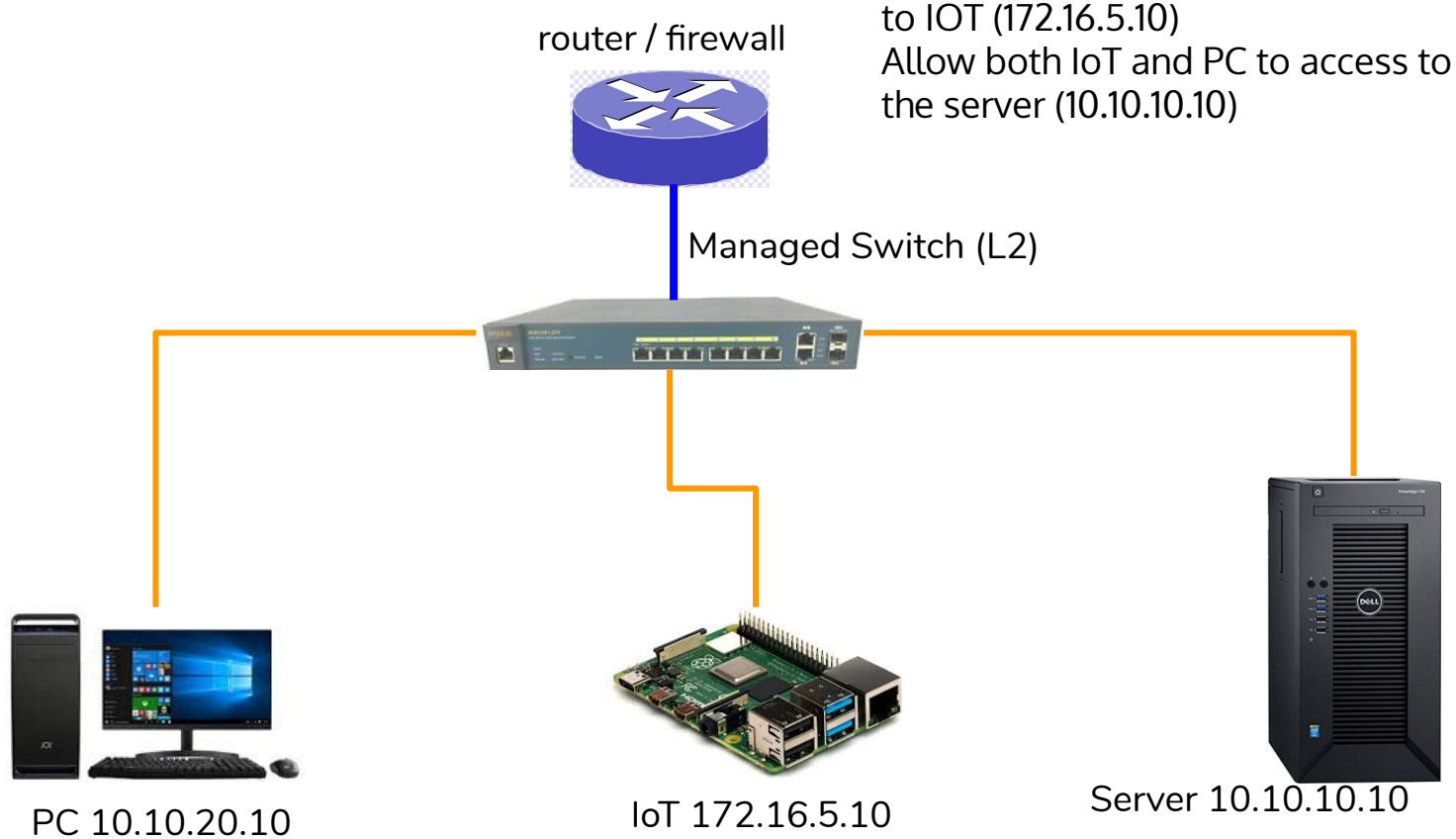
# 3.1 Network Design

## 3.1.4 Network Technology - IP Address (Inter VLAN Routing)

Steps:

1. Create VLANs in the main switch
2. Create trunking 802.1q for trunk link between router and main switch
3. Create interface for each VLAN in router / firewall
4. Assign IP address to VLAN interfaces accordingly to be network gateway
5. Create firewall rules as required

Firewall Rules:
Block access from PC (10.10.20.10) to IOT (172.16.5.10)
Allow both IoT and PC to access to the server (10.10.10.10)



# 3.1 Network Design

## 3.1.4 Network Technology - DHCP

A DHCP Server is a network server that automatically provides and assigns IP addresses, default gateways and other network parameters to client devices such as DNS servers.

It relies on the standard protocol known as Dynamic Host Configuration Protocol or DHCP to respond to broadcast queries by clients.

isc-dhcp on linux setup example

# 3.1 Network Design

## 3.1.4 Network Technology - DHCP

What is DHCP Scope:

- It is a range of IP addresses that a DHCP server can lease out to DHCP clients.
- This ensures no ip address duplicate in the network
- The IP addresses are leased for a specific Time to Live (TTL), usually three days. Information about scopes and leased IP addresses is stored in the DHCP database on the DHCP server.

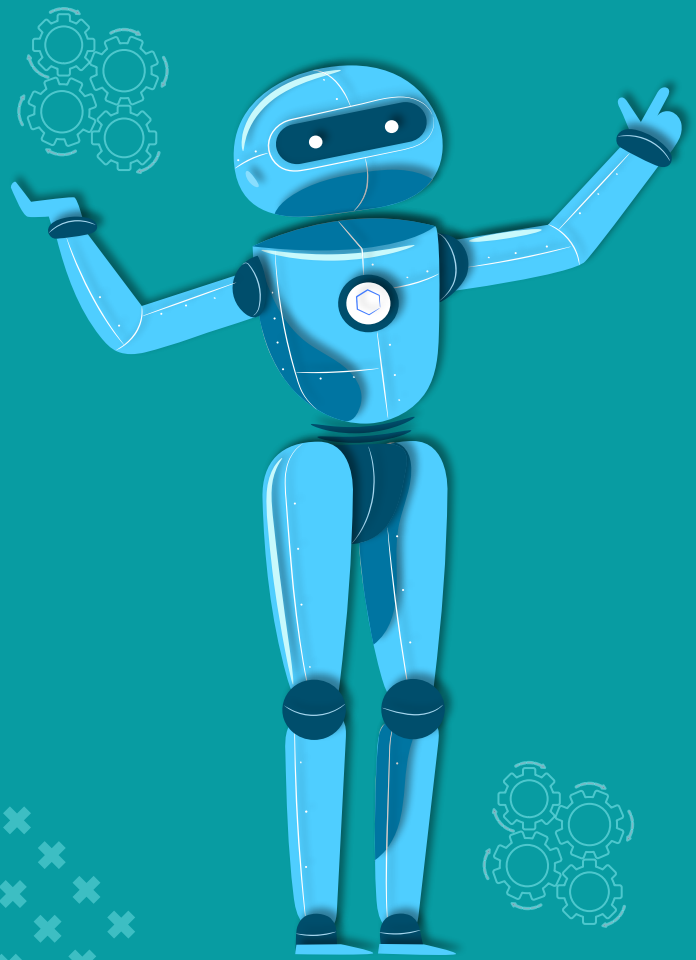

Ip addresses are pulled one by one  from the address pool consecutively and assigned to a new device that needs to connect to the network.

# 3.1 Network Design

## 3.1.4 Network Technology - DHCP

| | |
|---|---|
| Network ID | The network ID for the range of IP addresses |
| Subnet mask | The subnet mask for the network ID |
| IP address range | The range of IP addresses that are available to clients |
| Lease duration | The period of time that the DHCP server holds a leased IP address for a client before removing the lease |
| Router | A DHCP option that allows DHCP clients to Access remote networks. |
| Scope name | An alphanumeric identifier for administrative purposes. |
| Exclusion range | The range of IP addresses in the scope that is excluded from being leased. |

3.2



# Types of IoT Communication

- Bluetooth Low Energy
- Wi-Fi
- Cellular (4G, 4G LTE)
- Zigbee
- LoRaWAN
- NFC

# 3.2 Types of IoT Communication

## 3.2.1 Bluetooth Low Energy

Bluetooth Low Energy (BLE) is an enhanced version of Bluetooth, one of the oldest and most widely used wireless technologies for effective communication within the short range of approximately 10 meters.

The concept of Bluetooth was initiated by Nils Rydbeck at Ericsson Mobile in Sweden in 1989. Between 2001 and 2004, this was further optimized as lower power consumption and lower cost version Bluetooth Low Energy protocol or Bluetooth Smart by Nokia. It has been designed to offer significantly reduced power consumption while maintaining the communication range. Due to this attribute, Bluetooth is the leading protocol used by IoT devices. It is presently used by all major operating systems such as iOS, Android, Windows Phone, Blackberry, OS X, Linux, and Windows. The latest version of Bluetooth technology, Version 5.0, adds an innovative Internet Protocol Support Profile. It has been completely developed and optimized for Internet of Things devices.

WIP

# 3.2 Types of IoT Communication

## 3.2.2 Wi-Fi

WiFi is based on the IEEE 802.11 family of standards, and its first version was released in 1997. This version was capable of delivering up to 2Mbit/s link speeds.

Currently, the most common standard of WiFi is 802.11n, which is based on IEEE 802.11, but the use of 802.11ac is also rapidly increasing. The latest version provides even faster communication than 802.11n.

Although WiFi is a highly suited protocol for communication between IoT devices, it consumes high power for its operations. However, it is the most powerful protocol for file transfers among most of the IoT devices at present.

# 3.2 Types of IoT Communication

## 3.2.3 Cellular (4G, 4G LTE)

In March 2008, the ITU Radiocommunication Sector (ITU-R) released new standards for 4G ("Fourth Generation") connectivity, including faster connection speeds and mobile hotspots.


4G allows users to fully enjoy digital media on their mobile devices, including streaming video, rich multimedia apps, and high-quality music. Users can start watching a movie in seconds, without worrying about long load times and buffering. When 4G was first introduced, it was more of a hypothetical target for tech developers, and most carriers didn't support the new network to its full potential. Today, 4G is the new standard, unless you're in a dead zone where 3G is the only available option.

4G LTE is a major improvement over 3G speeds, it is technically not 4G. It is an improvement over its predecessor, but not substantial enough to qualify as a new generation. It is a clever workaround that allows cellular networks to advertise 4G speeds, without reaching the minimum standards set by the ITU-R.

WIP

# 3.2 Types of IoT Communication

## 3.2.4 Zigbee

ZigBee is a short-range wireless communication protocol based on IEEE 802.15.4 standard and operates on the frequency of 2.4GHz with a 250kbps data rate. The major attributes that make ZigBee suitable for effective communication between IoT devices are low power consumption, high scalability, security, and durability along with high node counts. While the maximum number of nodes in the network can be 1024 with a range of up to 200 meters, ZigBee can even use 128 bit AES encryption.

ZigBee protocol is ideally designed for use in home automation and big industrial sites where low power is required, and data exchange among home or building is infrequent at low data rates. There is a vast range of user base using ZigBee as the preferred mode of communication between IoT devices.

# 3.2 Types of IoT Communication

## 3.2.5 LoRaWAN

Long Range Wide Area Network (LoRaWAN) is a protocol that is primarily intended for long-range wireless battery-operated IoT devices in regional, national, or global networks. It is specifically known for its ability to communicate in long-range with the least power consumption and detects the signals below the noise level. This protocol is mainly used in smart cities, where there is a large network with millions and millions of devices connected to each other that function with less power and memory, low-cost mobile secure communication in IoT devices, and a wide range of industrial applications. It has a data rate of 0.3kbps to 50kbps.


WIP

# 3.2 Types of IoT Communication

## 3.2.6 NFC

NFC stands for “Near Field Communication” and, as the name implies, it enables short-range communication between compatible devices.

NFC technology has its roots in RFID tags, first developed in 1983, which consumers may know from its applications in retail or inventory tracking.

Over time, NFC technology has become smaller and c popular as the technology
Pay and Samsung Pay.

Passive NFC devices include tags, and other small transmitters, that can send information to other NFC devices without the need for a power source of their own.

The transmission frequency for data across NFC is 13.56 megahertz. You can send data at either 106, 212, or 424 kilobits per second.

Active devices are able to both send and receive data, and can communicate with each other as well as with passive devices. Smartphones are by far the most common form of active NFC device. Public transport card readers and touch payment terminals are also good examples of the technology.

WIP

# 3.2 Types of IoT Communication

## 3.2.6 NFC

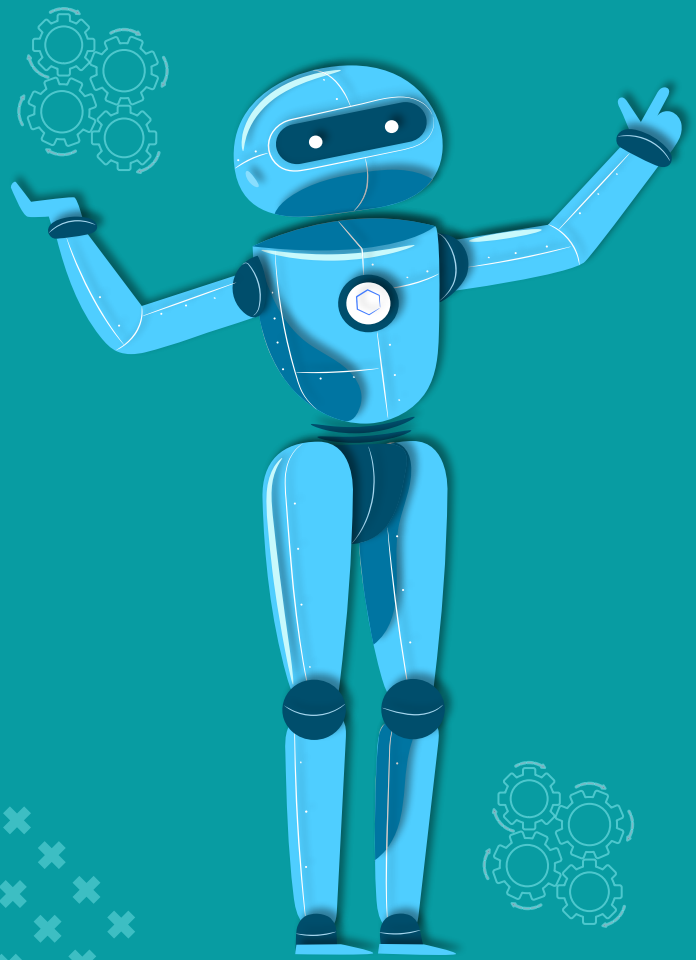
Active devices are able to both send and receive data, and can communicate with each other as well as with passive devices. Smartphones are by far the most common form of active NFC device. Public transport card readers and touch payment terminals are also good examples of the technology.

Passive NFC devices include tags, and other small transmitters, that can send information to other NFC devices without the need for a power source of their own.

The transmission frequency for data across NFC is 13.56 megahertz. You can send data at either 106, 212, or 424 kilobits per second.

WIP

3.3



# Virtualization and Cloud Platform

- Underlying Technology - Cloud and Virtualization
- Virtualization Technology
- Virtual Box
- Docker

# 3.3 Virtualization and Cloud Platform

## 3.3.1 Underlying Technology - Cloud and Virtualization

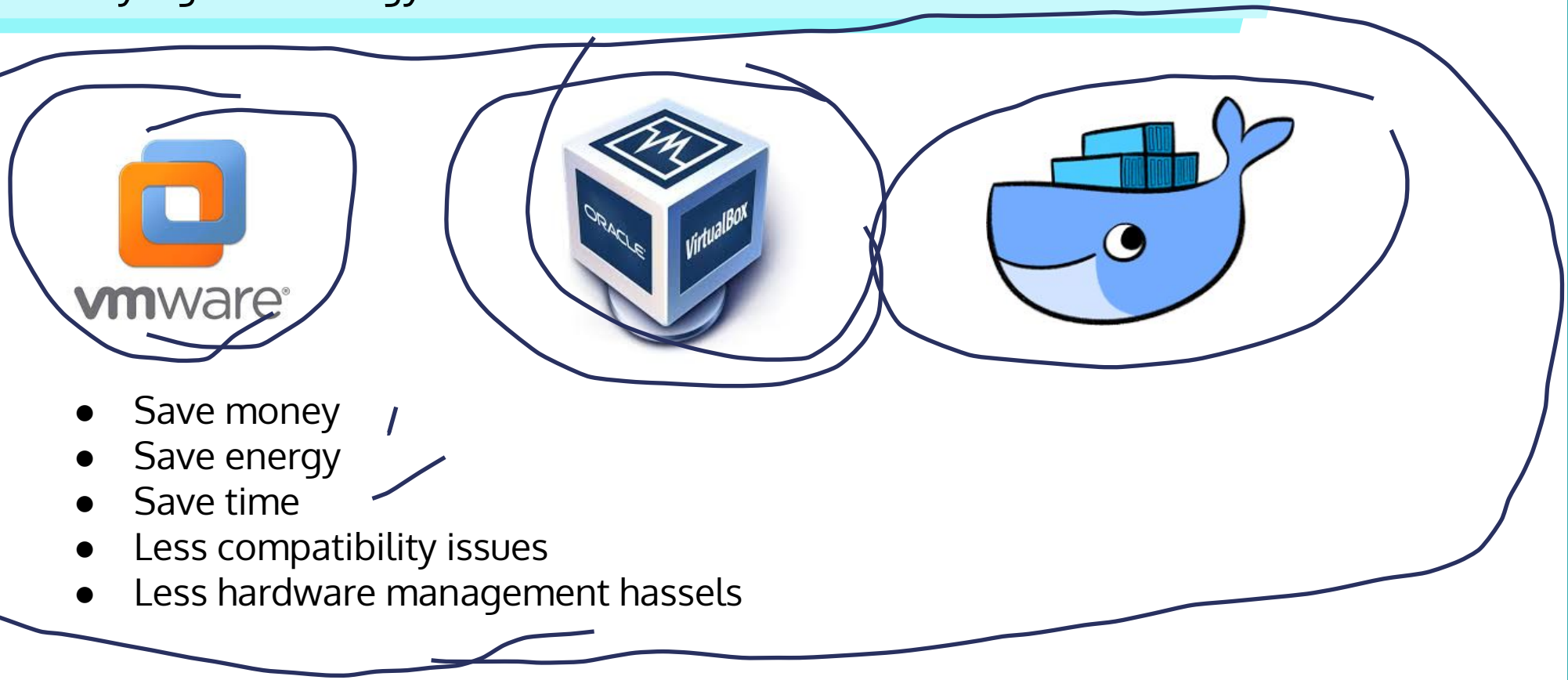
| | |
|---|---|
| **SaaS** | Google Apps, Dropbox, Salesforce, Cisco WebEx, Concur, GoToMeeting |
| **PaaS** | AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos, OpenShift |
| **IaaS** | DigitalOcean, Linode, Rackspace, Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure, Google Compute Engine (GCE) |



WIP

# 3.3 Virtualization and Cloud Platform

## 3.3.1 Underlying Technology - Cloud and Virtualization




- Save money
- Save energy
- Save time
- Less compatibility issues
- Less hardware management hassels

# 3.3 Virtualization and Cloud Platform

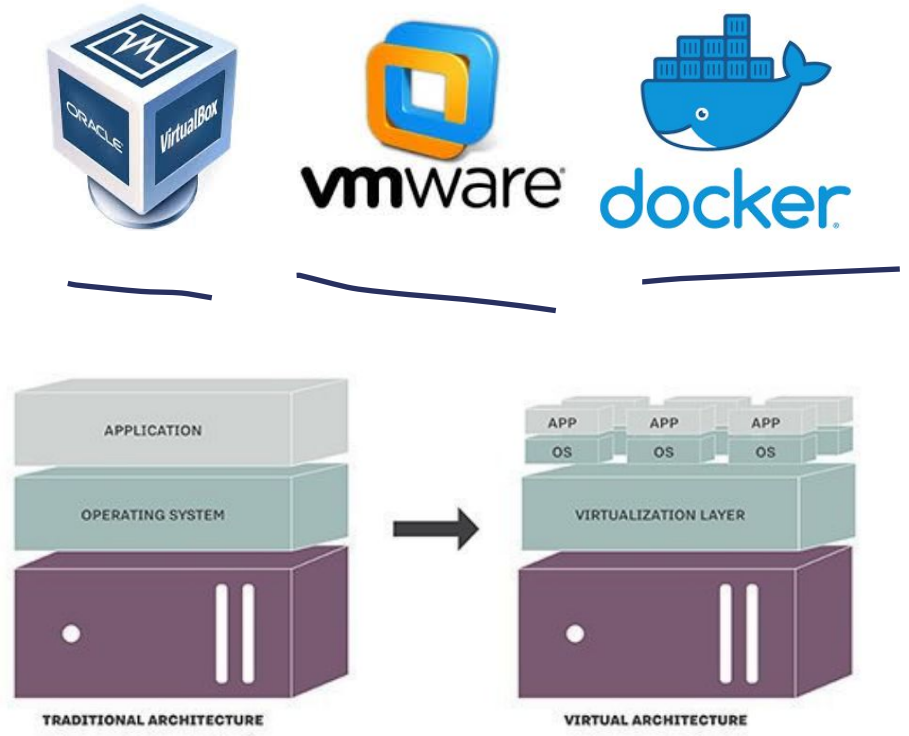
## 3.3.2 Virtualization Technology

1. Server virtualization concept
2. Application containerization concept

Server virtualization is a virtualization technique that involves partitioning a physical server into a number of small, virtual servers with the help of virtualization software.

In server virtualization, each virtual server runs multiple operating system instances at the same time.

The concept of server virtualization is widely applied in IT infrastructure as a way of minimizing costs by increasing the utilization of existing resources

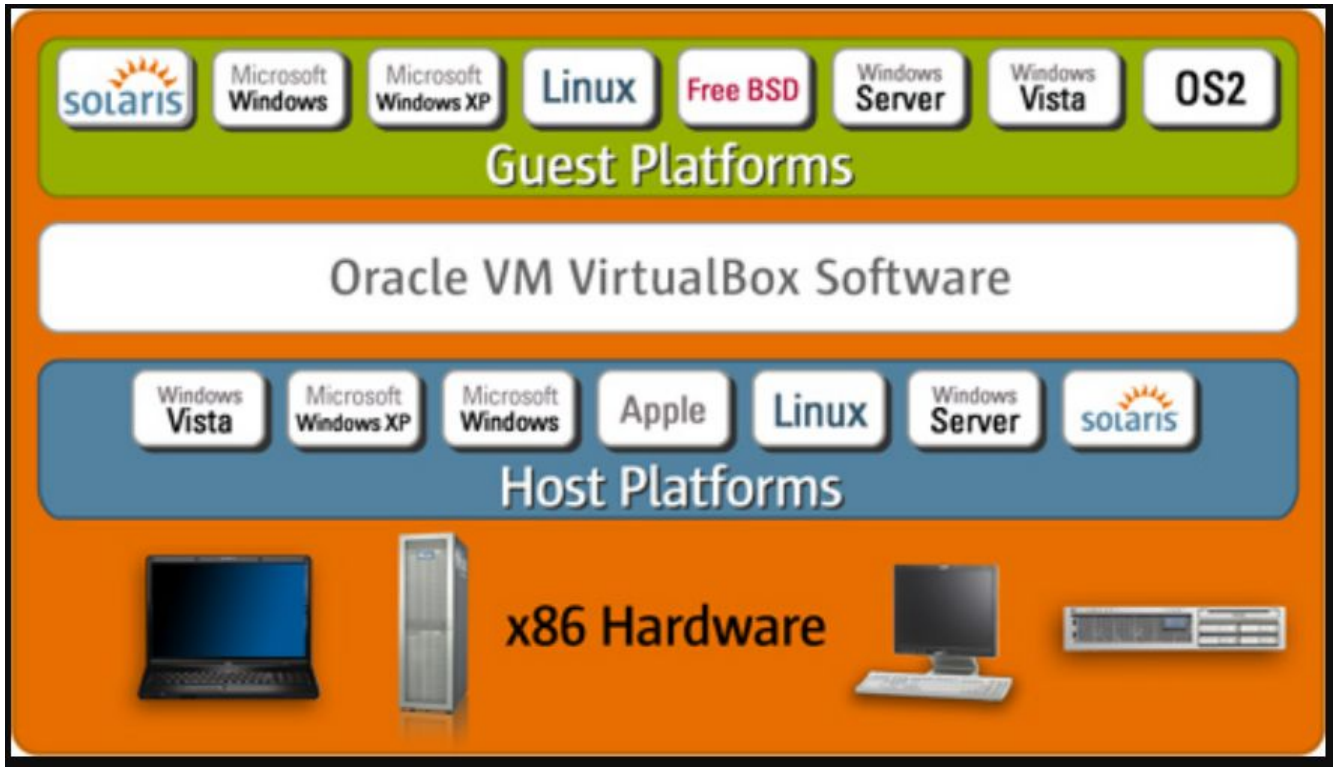


# 3.3 Virtualization and Cloud Platform

## 3.3.2 Virtualization Technology

Oracle VM VirtualBox ("VirtualBox") is a cross-platform virtualization engine for use on computers running Microsoft Windows, the most popular Linux distributions, Oracle Solaris, or MacOS.

Designed for use on Intel and AMD x86 systems, Oracle VM VirtualBox can be deployed on desktop or server hardware. As a hosted hypervisor, it extends the existing operating system installed on the hardware rather than replacing it.

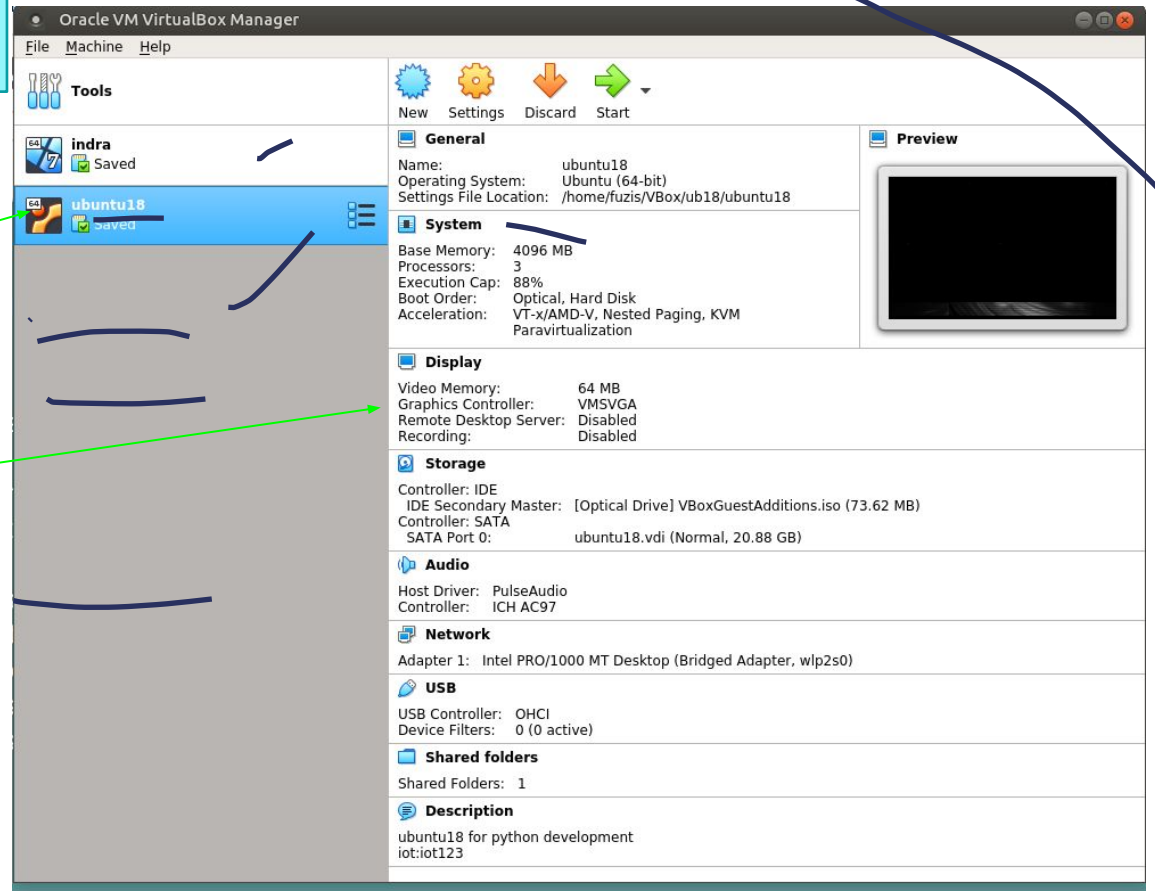


# 3.3 Virtualization and Cloud Platform

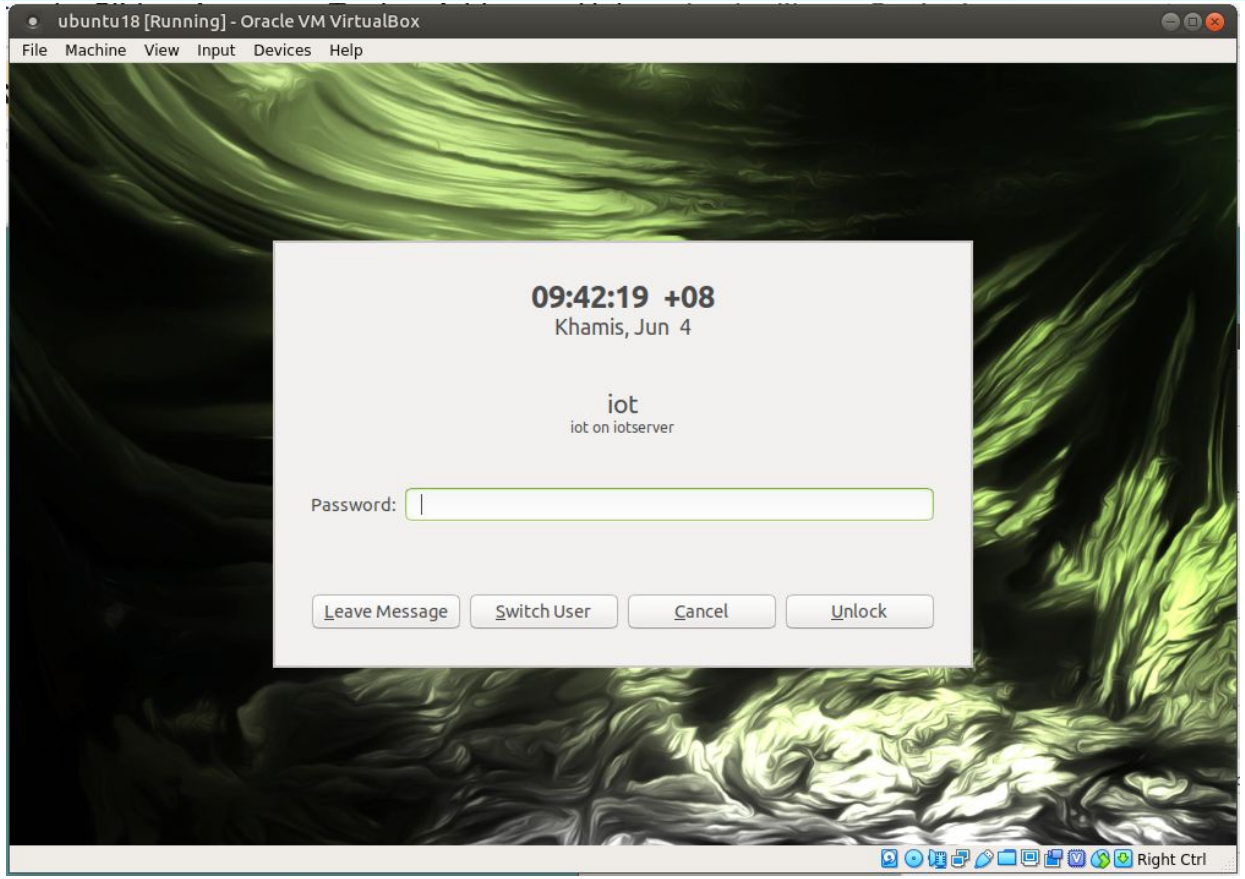
## 3.3.3 VirtualBox - Oracle VM VirtualBox (software)

Virtual machine

Virtual machine setup

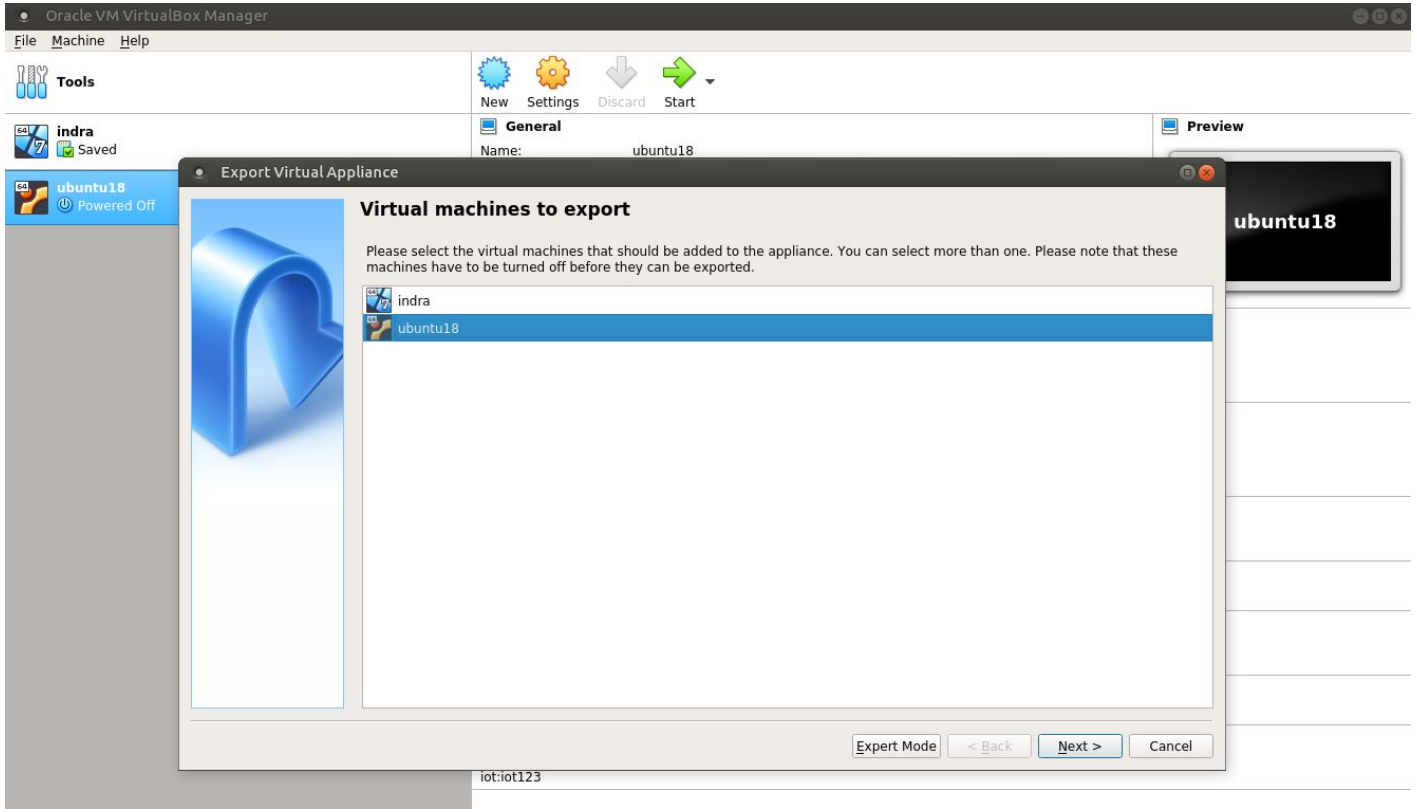


## 3.3.3 VirtualBox - Example Ubuntu desktop OS running in VirtualBox



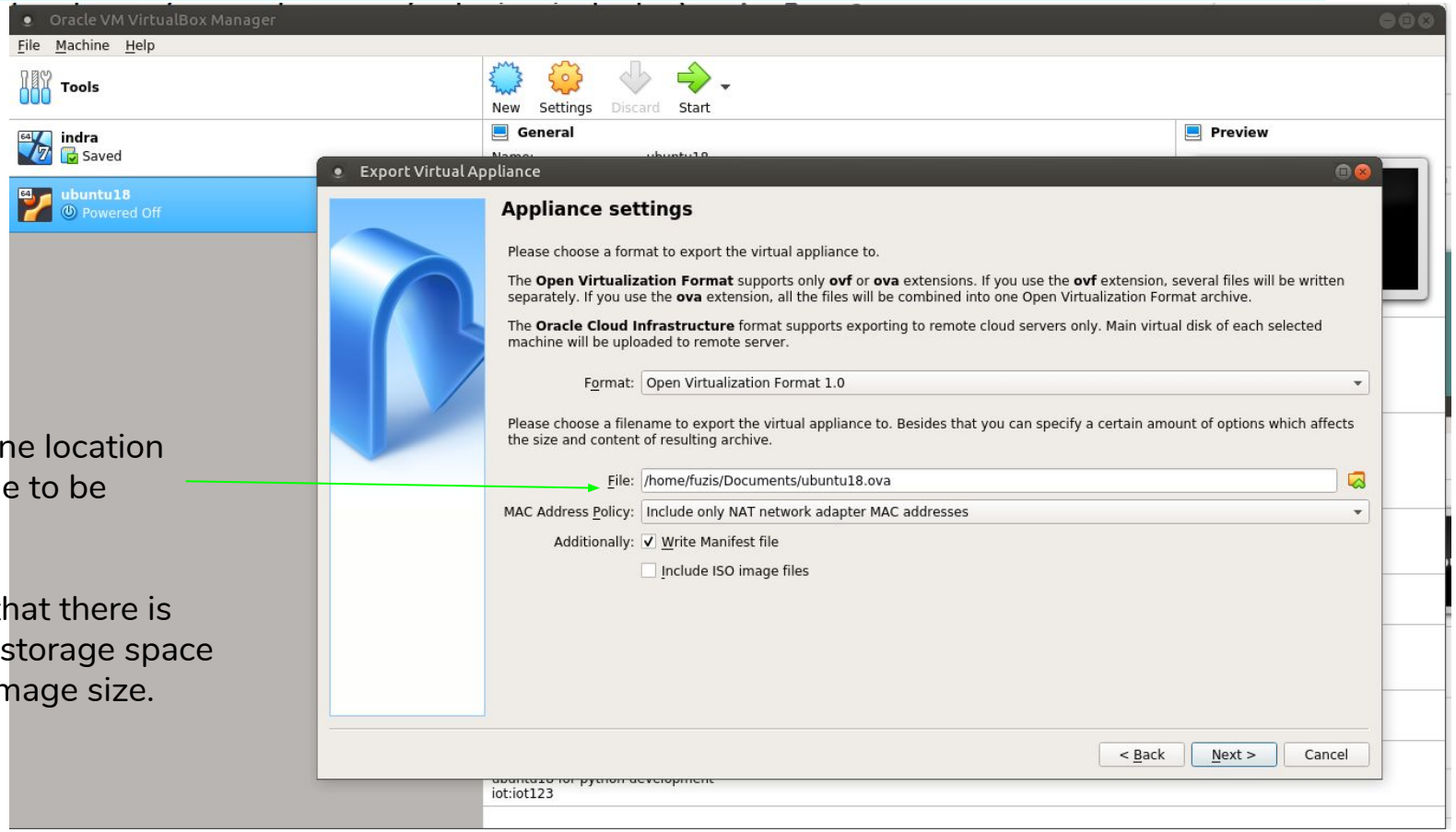
# 3.3 Virtualization and Cloud Platform

## 3.3.3 VirtualBox - export a virtual machine instance as backup



# 3.3 Virtualization and Cloud Platform

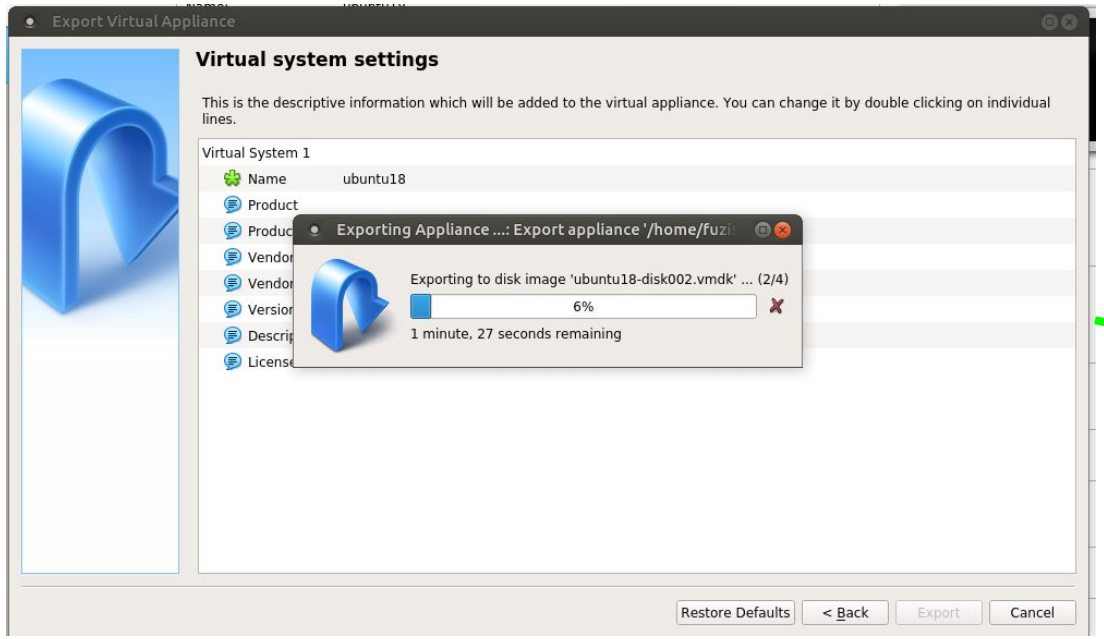
## 3.3.3 VirtualBox - export / backup virtual machine appliance image file location




Determine location for image to be copied

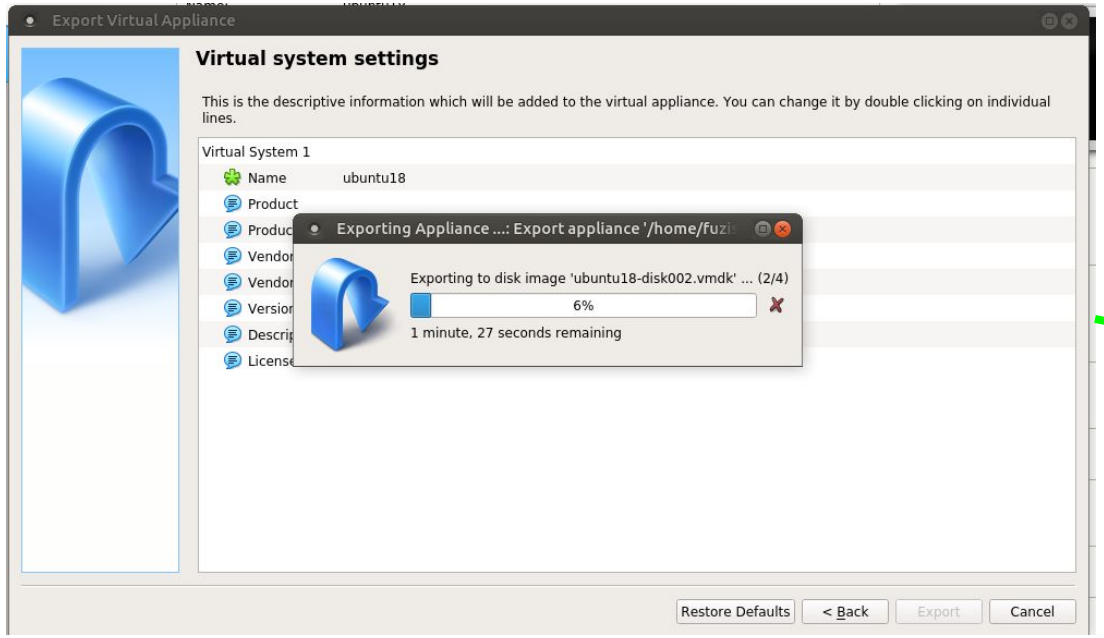
Ensure that there is enough storage space for the image size.

## 3.3.3 VirtualBox - export / backup progress

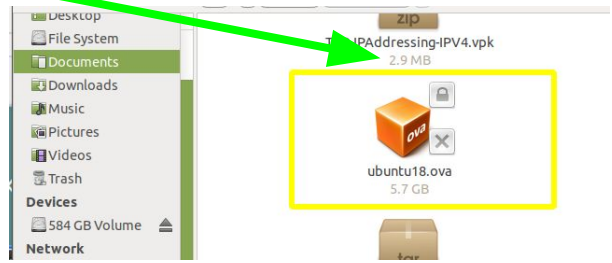



completed export backup appliance image

## 3.3.3 VirtualBox - export / backup progress

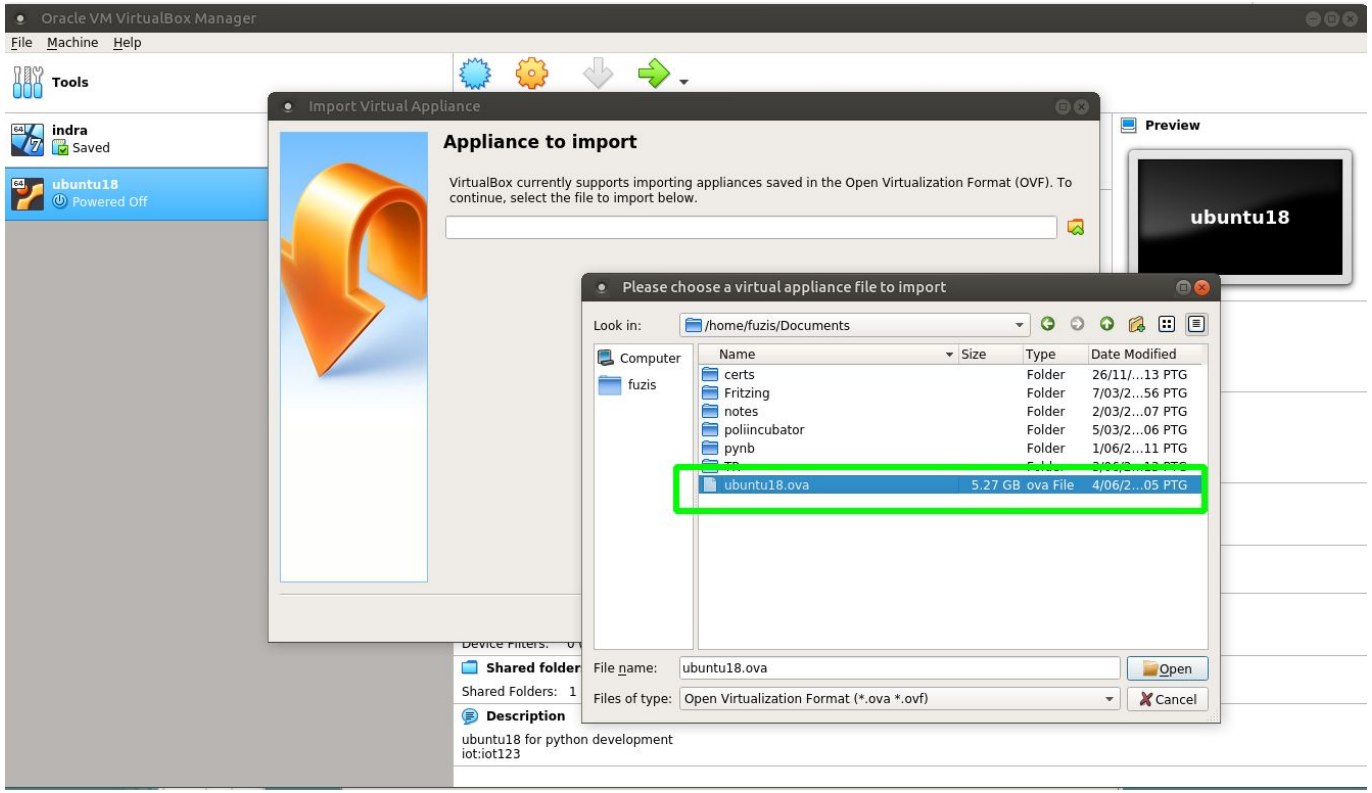



completed export backup appliance image



## 3.3.3 VirtualBox - Import appliance

Appliance file can be imported by virtualbox program to restore the server image or to redeploy / migrate to another machine.



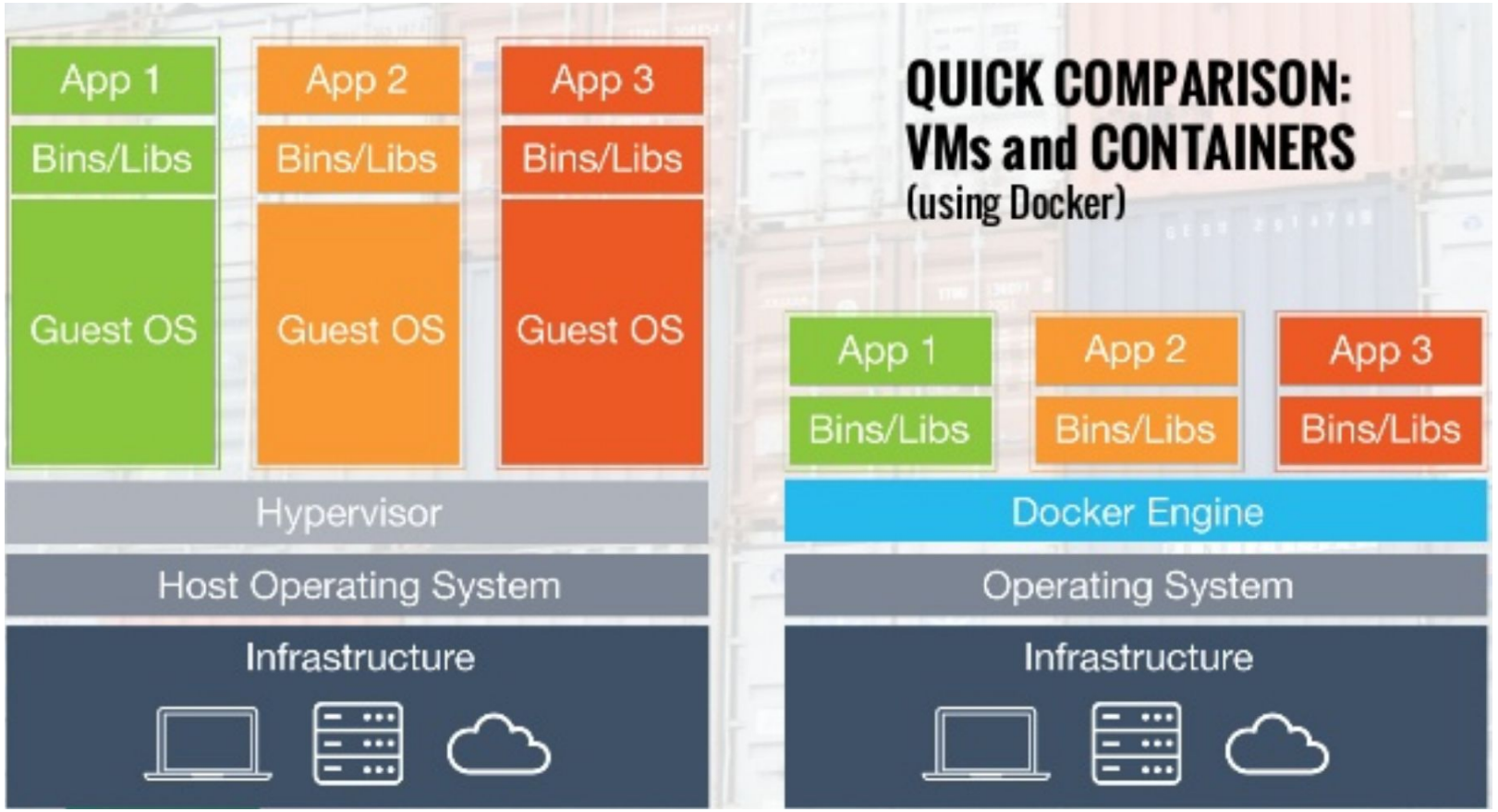
## 3.3.4 Docker - What is Docker?

- Docker is an open-source project that automates the deployment of software applications inside containers by providing an additional layer of abstraction and automation of OS-level virtualization on Linux. -- wikipedia

  What are containers?

- The industry standard today is to use Virtual Machines (VMs) to run software applications. VMs run applications inside a guest Operating System, which runs on virtual hardware powered by the server's host OS.
- Containers leverages the low-level mechanics of the host operating system, containers provide most of the isolation of virtual machines at a fraction of the computing power, hence this frees up hardware resources used for creating virtualized hardware like virtual machines do.

## 3.3.4 Docker - Docker and Virtual Machine Comparison

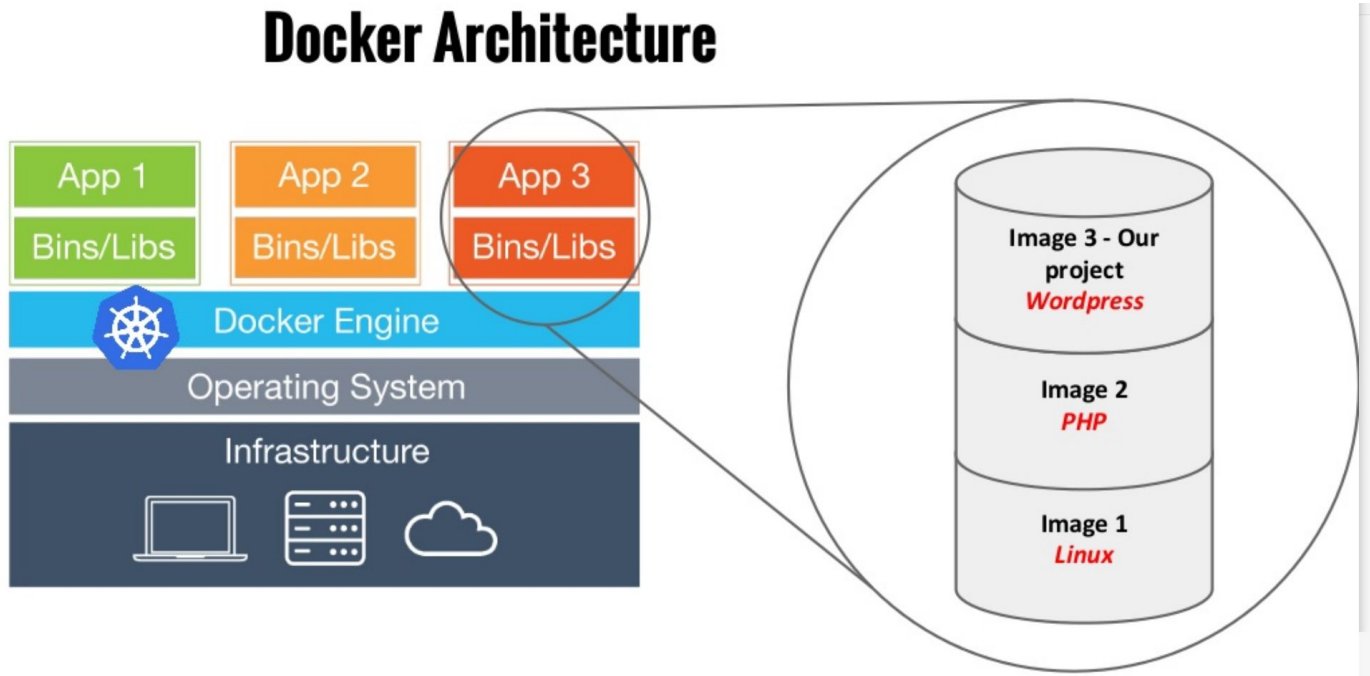


## 3.3.4 Docker

Docker eliminates hypervisor engine and allow the container to use host OS service to support application in containers.

Disadvantage is that, that application built must be from same type of OS running the docker engine.

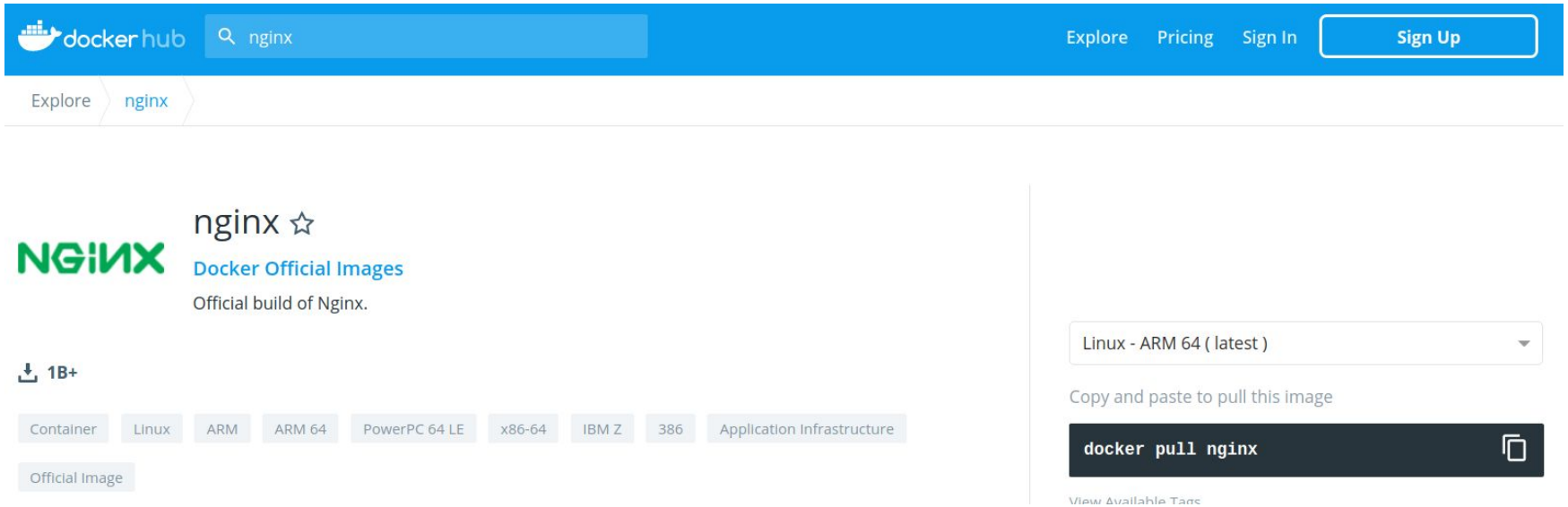
**installation direction**



## 3.3.4 Docker - Docker Image

Docker images are available for various application, they need to be pull from the repository, https://hub.docker.com/

We need an internet access to do this



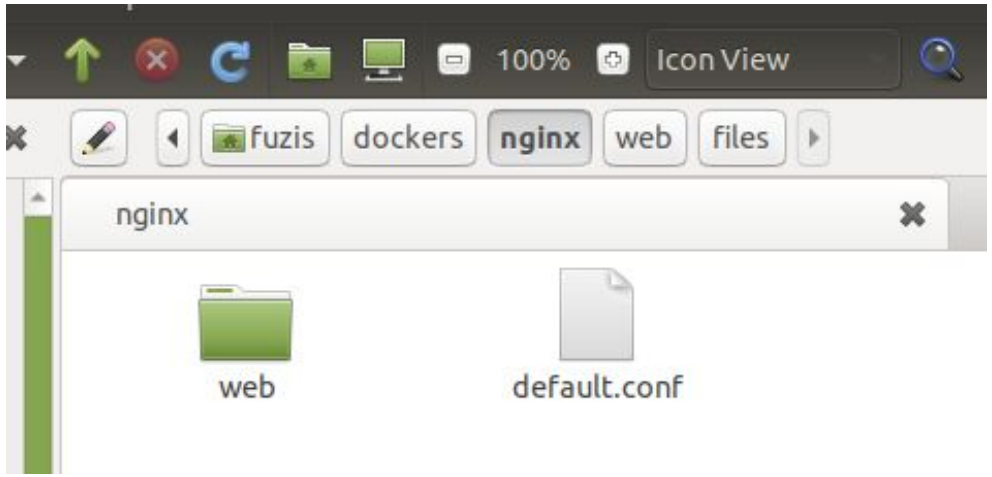
# 3.3 Virtualization and Cloud Platform

## 3.3.4 Docker - Docker Image

Let's pull Nginx image and run as a web server to publish our static contents on the network.

Steps:

1. Create a folder name "nginx"
2. Create another folder called "web" inside the folder nginx
3. Edit create and edit default.conf, add autoindex on;
4. sudo docker run -d -p 80:80 -v /home/fuzis/dockers/nginx/default.conf:/etc/nginx/conf.d/default.conf -v /home/fuzis/dockers/nginx/web:/usr/share/nginx/html:ro nginx
5. Create subfolder in the web directory and any files copied there will be published over the network via web server Nginx.



# 3.3 Virtualization and Cloud Platform

## 3.3.4 Docker - Docker run image Nginx example

```
docker pull nginx
```

```
File  Edit  View  Search  Terminal  Help
(base) fuzis@fsvivo:~$ sudo docker image ls
REPOSITORY                                  TAG                 IMAGE ID            CREATED             SIZE
wordpress                                   latest              b301a17258fe        7 days ago          540MB
nginx                                       latest              6678c7c2e56c        3 months ago        127MB
```

```
#>sudo docker run -d -p 80:80 -v /home/fuzis/dockers/nginx/default.conf:/etc/nginx/conf.d/default.conf -v /home/fuzis/dockers/nginx/web:/usr/share/nginx/html:ro nginx
```

```
(base) fuzis@fsvivo:~/dockers/nginx$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                  NAMES
e6ad1206cf5e        nginx               "nginx -g 'daemon of…"   2 minutes ago       Up 2 seconds        0.0.0.0:80->80/tcp     gallant_easley
(base) fuzis@fsvivo:~/dockers/nginx$
```

check docke images running...

# 3.3 Virtualization and Cloud Platform

## 3.3.4 Docker - Docker run image switches explanation

```
#>sudo docker run -d -p 80:80 -v /home/fuzis/dockers/nginx/default.conf:/etc/nginx/conf.d/default.conf -v /home/fuzis/dockers/nginx/web:/usr/share/nginx/html:ro nginx
```

| | |
|---|---|
| sudo docker run | running a docker  image. If the image is not available yet in the local repository, it will be download from docker hub. |
| -d | docker container runs in detached or daemon mode. It will run in the background and you can use command “docker ps” to see the status. |
| -p | port mapping, docker will forward host port to container port ***<host port>:<container port>***. This will make the container accessible from the local network from other hosts or PCs. |
| -v | volume mapping, docker will map ***<host directory>:<container directory>,*** this make container to be able to access directory contents on the host PC or server and use it in the container application. This will make the data or files modified by the application persisted even though the container is deleted. |

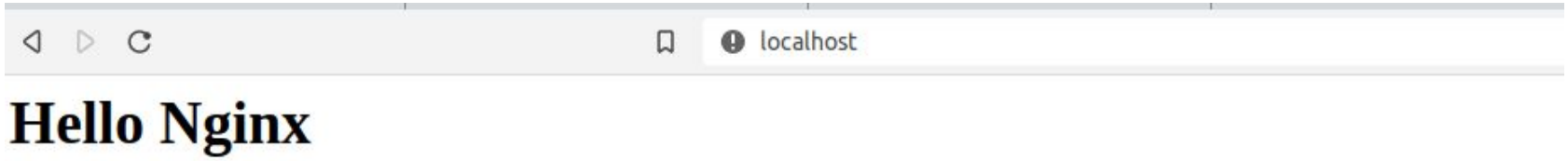
# 3.3 Virtualization and Cloud Platform

## 3.3.4 Docker - Docker run image Nginx example


index.html

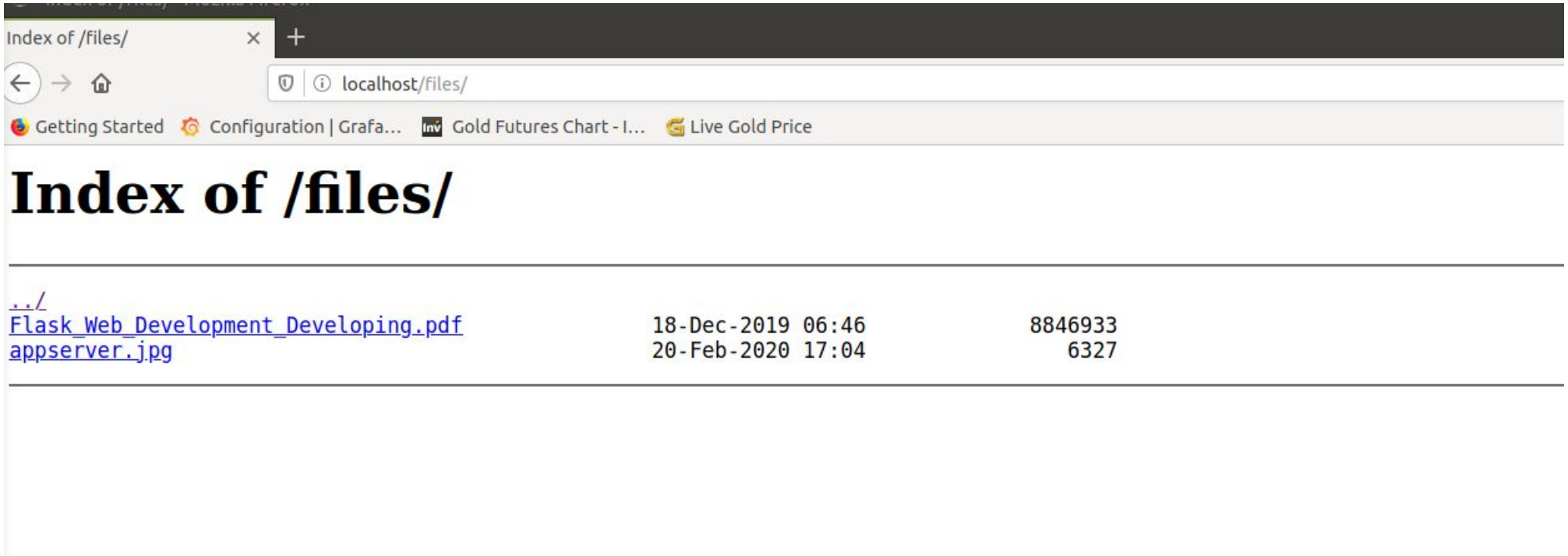
```
<html>
<head></head>
<body>
<h1>Hello Nginx </h1>
</body>

</html>
```



## 3.3.4 Docker - Docker Image

Nginx docker image web server publish our static contents on the network.



## 3.3.4 Docker - Dockerfile

### What is a Dockerfile ?

There is always a scenario that you need to build your app differently from the standard image published on docker hub. Such as, your app directory will be different name like restapp than just “app” or change the OS base to different distribution.  Then you need to use Dockerfile to build your own image first by using command docker build. Then you can run that image as usual.

*A Dockerfile is a text file that defines a Docker image.*

Dockerfile example

```
FROM python:3
WORKDIR /usr/src/app
RUN apt-get update
COPY requirements.txt ./
RUN pip install --no-cache-dir -r
requirements.txt

COPY src/ /usr/src

CMD [ "python", "/usr/src/app/app.py" ]
```

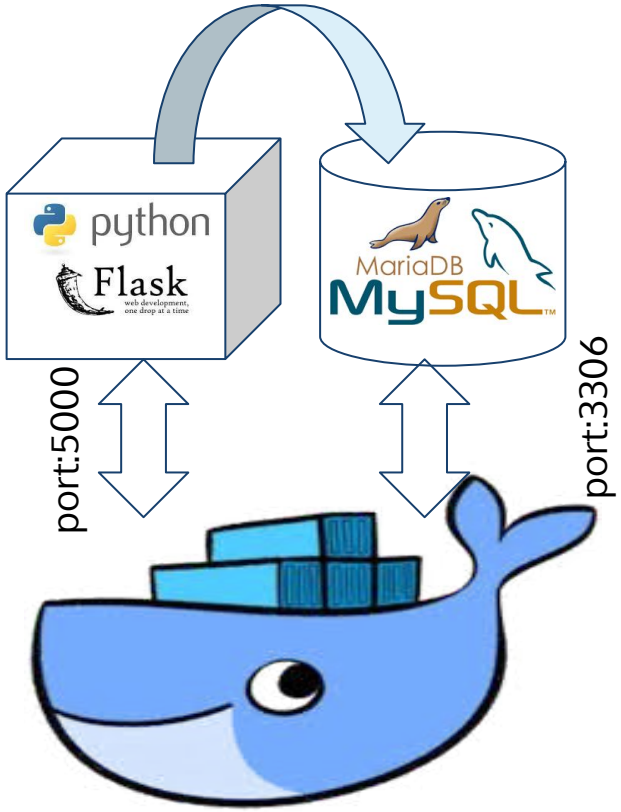
## 3.3.4 Docker - Compose

### What is Docker Compose ?

Docker Compose is the toolkit provided by Docker to build, ship and run multi-container applications.

What does it means....?

It means that, you can use this tool when you need to use more than one container images that will work together in you application, such as flask and MySQL.




**Note:**

*Docker build its own virtual network to let its docker images communicate with each other. But you do not need to assign them unique ip addresses in order to let them identify each other, but, Docker used unique id name defined in docker-compose.yml under key "link" example:*

```
links:
  - db
```

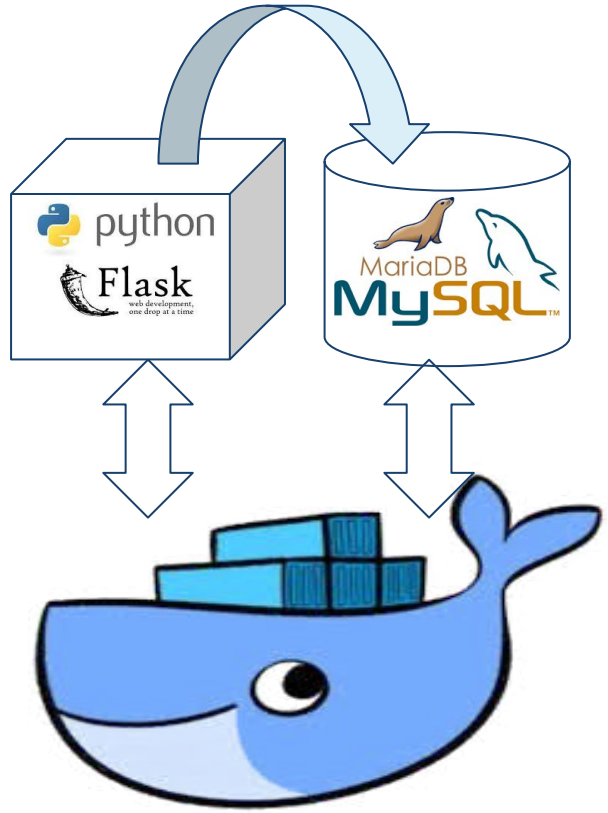
## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

**STEPS:**

1. Create a project directory such as "/flaskapp"
2. Create another directory for flask working directory, such as "/app"
3. Create a "Dockerfile" inside the flask working directory ("/app")
4. Create a "requirements.txt", a file that will tell python package installer (pip) what the dependencies for your flask app, such as ORM and mysql driver libraries.
5. create a folder for MySQL data directory inside the project directory ("/flaskapp"). It will be used by MySQL to store data files. Let's name it "/db"
6. Create a ***"docker-compose.yml"*** inside the project directory.
7. copy your flask application code into the flask working directory "/app"
8. execute command ***#>docker-compose up***

After docker-compose up is executed, it will read the Dockerfile which contains instructions to search the repository for python images, and then execute "pip" inside the container to install flask libraries and the dependencies. The same happens for mySQL, in most cases the docker MySQl standard image is sufficient, no need for customization.



## 3.3.4 Docker - Compose

### What do you need and how to do it… ?

Below is the content if our **"Dockerfile"** file:

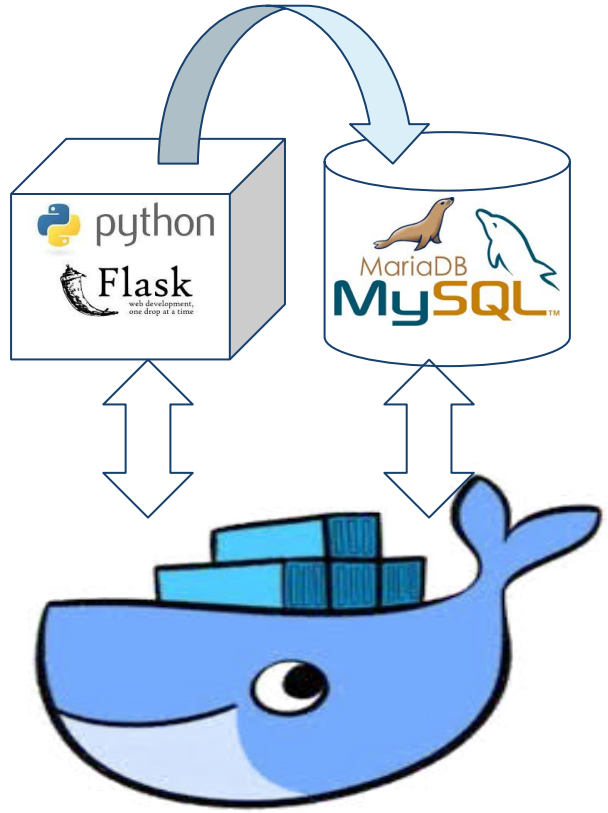
```
# Use an official Python runtime as an image
FROM python:3.6

# The EXPOSE instruction indicates the ports on which a container
# will listen for connections
# Since Flask apps listen to port 5000  by default, we expose it
EXPOSE 5000

# Sets the working directory for following COPY and CMD instructions
# Notice we haven't created a directory by this name - this instruction
# creates a directory with this name if it doesn't exist
WORKDIR /app

# Install any needed packages specified in requirements.txt
COPY requirements.txt /app
RUN pip install -r requirements.txt

# Run app.py when the container launches
COPY app.py /app/app.py
CMD python app.py
```

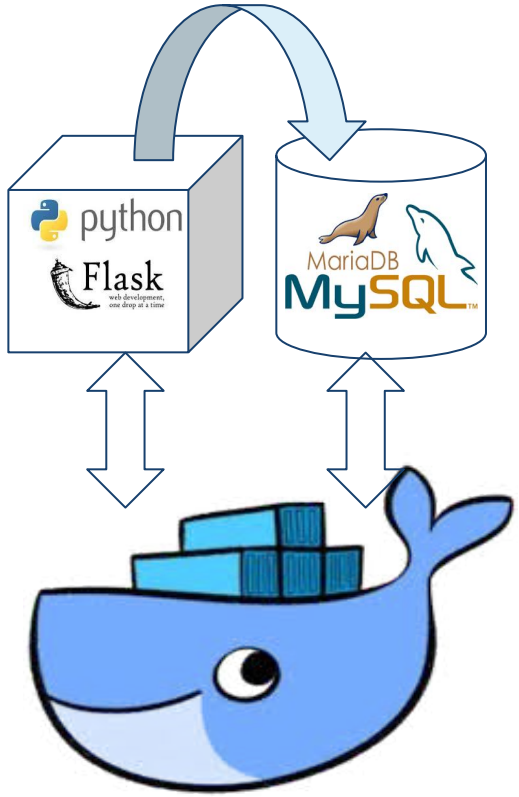


## 3.3.4 Docker - Compose

### What do you need and how to do it… ?

Below is the content of **"requirements.txt"** file:

```
quart
flask
peewee
flask-peewee
PyMySQL
flask-restful
```




requirements.text contains the list of dependencies or libraries that your flask app is going to need. It will be used by python installation program (pip) to install them in the docker container during the building process.

quart and flask are the python micro web frameworks that can be used to build a web application and both of them support REST protocols. Coding with them is easy and fun!

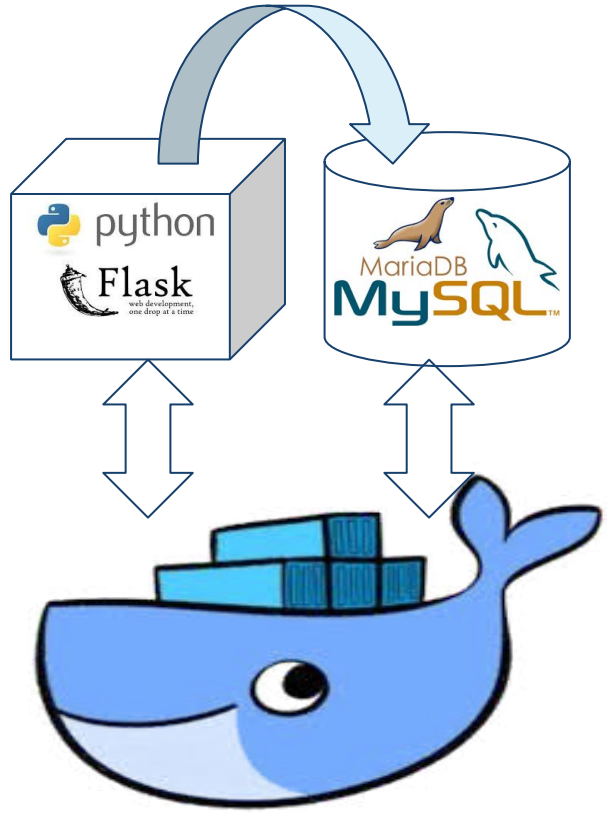
peewee, flask-peewee are the libraries to support ORM codes.
PyMySQL is the driver for python to communicate with MySQL database.
flask-restful is the library to create REST compliant codes.

## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

Below is the content if our **docker-compose.yml** file

```
version: "3.3"
services:
  app:
    build: ./app
    container_name: "Flask_App"
    links:
      - db
    ports:
      - "5000:5000"
  db:
    image: mysql:5.7
    container_name: "MySQL_DB_flask"
    ports:
        - "3306:3306"
    environment:
        MYSQL_ROOT_PASSWORD: rootpwd
        MYSQL_DATABASE: "iot"
    volumes:
        - ./db:/docker-entrypoint-initdb.d/:rw
```



## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

```
from flask import Flask
from flask import jsonify, request
from datetime import datetime
from peewee import *
myDB = MySQLDatabase("iot", host="db",
port=3306, user="root", passwd="rootpwd")
myDB.connect()

class MySQLModel(Model):
        class Meta:
                database = myDB

class DHT22(MySQLModel):
   timestamp = TimestampField()
   IOT = CharField()
   temperature = FloatField()
   humidity = FloatField()

myDB.create_tables([DHT22,])
app = Flask(__name__)
```

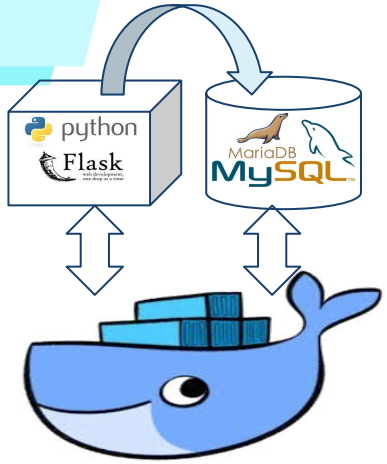
Below is the content if our flask app.py file:

```
def tstam():
   return datetime.now().timestamp()

@app.route('/')
def test():
   return jsonify({'Hello': 'IOT !'})

@app.route('/log', methods=['POST'])
def sens():
   j = request.json
   print(j)
   dht = DHT22()
   dht.IOT = j["IOT"]
   dht.temperature = j["temperature"]
   dht.humidity = j["humidity"]
   dht.save()
   return "IOT data arrived... ! {}".format(request.json)

if __name__ == '__main__':
   app.run(host='0.0.0.0',port=5000, debug=True)
```



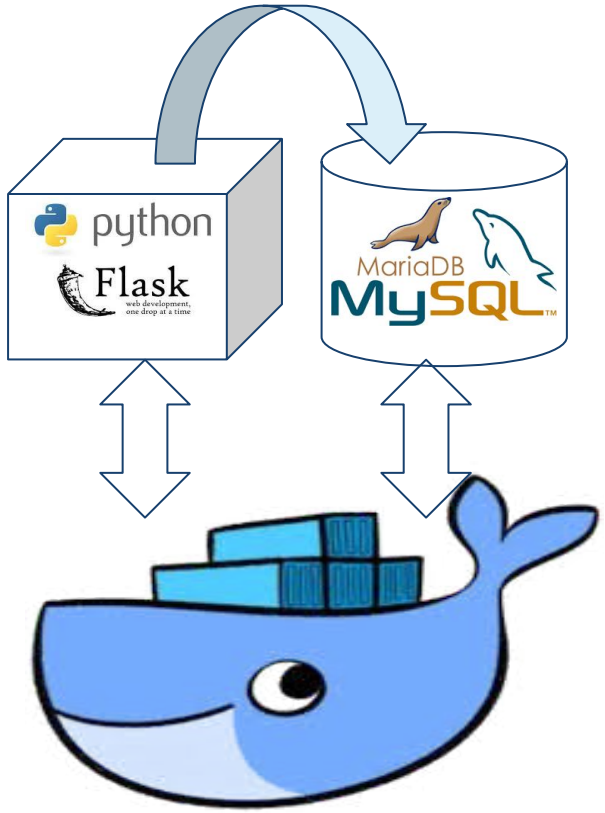
## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

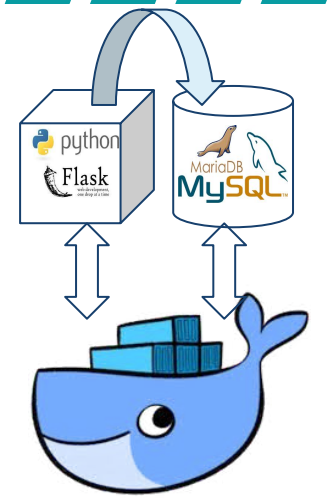
Below is the directory structure:

```
(base) fuzis@fsvivo:~/flaskapp$ tree
.
├── app                    <- Working directory
│   ├── app.py             <- flask code
│   ├── Dockerfile
│   └── requirements.txt   <- dependencies list for python installer
├── db                     <- MySQL storage directory
└── docker-compose.yml
```





## 3.3.4 Docker - Compose




**What do you need and how to do it… ?**

Executing docker-compose up:

```
File  Edit  View  Search  Terminal  Help
(base) fuzis@fsvivo:~/flaskapp$ sudo docker-compose up
Creating network "flaskapp_default" with the default driver
Creating MySQL_DB_flask ... done
Creating Flask_App      ... done
Attaching to MySQL_DB_flask, Flask_App
MySQL_DB_flask | 2020-06-13 17:43:04+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.
7.30-1debian10 started.
MySQL_DB_flask | 2020-06-13 17:43:04+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
MySQL_DB_flask | 2020-06-13 17:43:04+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.
7.30-1debian10 started.
MySQL_DB_flask | 2020-06-13 17:43:04+00:00 [Note] [Entrypoint]: Initializing database files
MySQL_DB_flask | 2020-06-13T17:43:04.186302Z 0 [Warning] Changed limits: max_open_files: 1024 (reques
```

Check with docker ps command:

*Note: you can use "docker start <image ID>" manually if they do not start automatically, and the right order will be database server then the flask app.*

```
(base) fuzis@fsvivo:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                               NAMES
a383cdb680bc        flaskapp_app        "/bin/sh -c 'python …"   12 minutes ago      Up 31 seconds       0.0.0.0:5000->5000/tcp              Flask_App
728bfda10b1a        mysql:5.7           "docker-entrypoint.s…"   12 minutes ago      Up 12 minutes       0.0.0.0:3306->3306/tcp, 33060/tcp   MySQL_DB_flask
(base) fuzis@fsvivo:~$
```

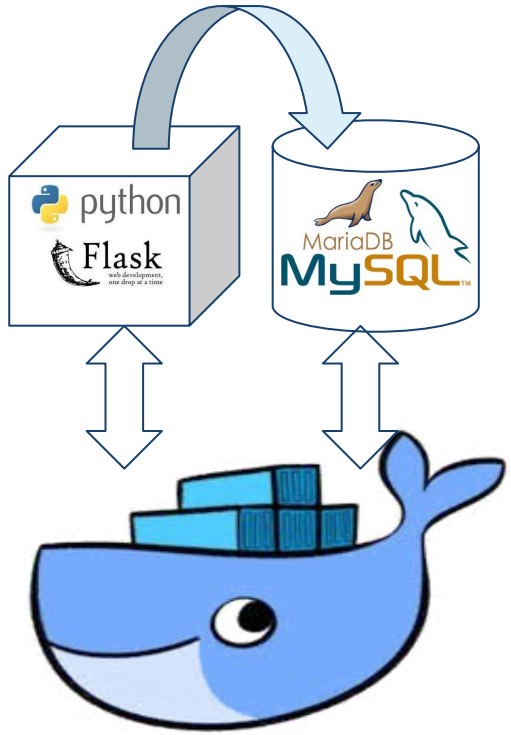
## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

Accessing the application:

1. You can use mysql Workbench or any MySQL dbms tool to access the database.
2. Use a browser to check the flask application web access.
3. Use Postman or any REST compliant tool to check flask app REST endpoints.

Below is the screenshot of testing access to flask app using a web browser:

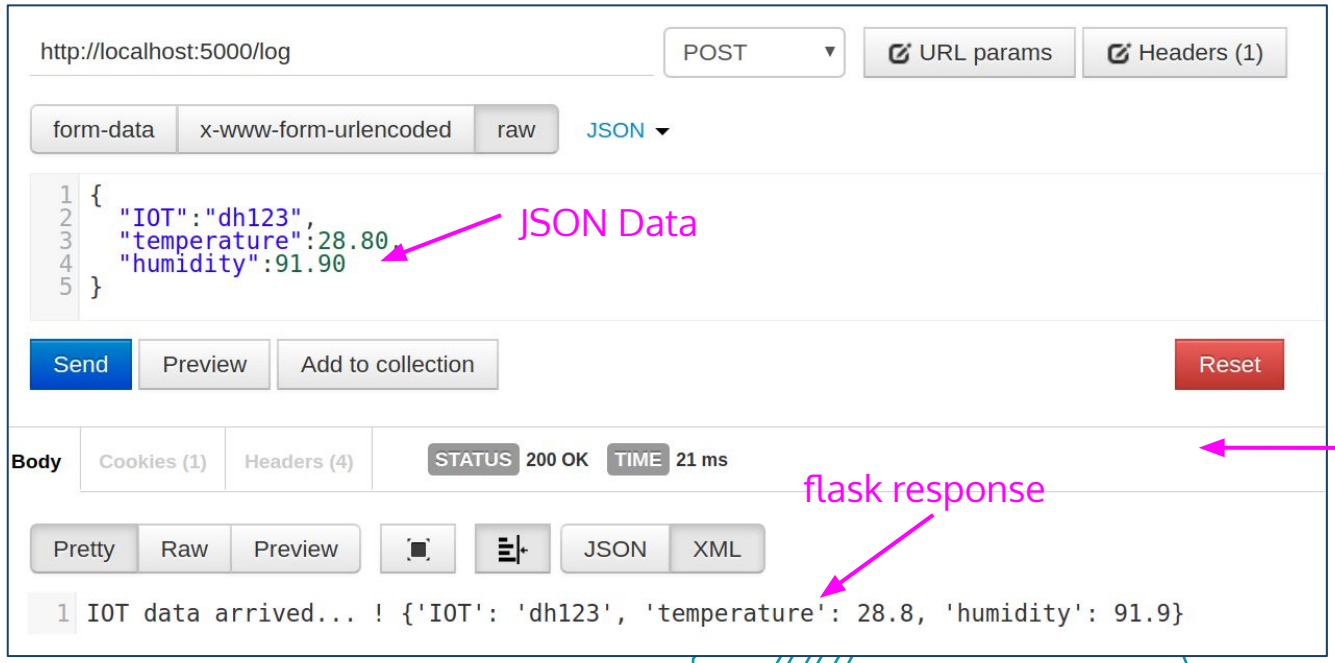



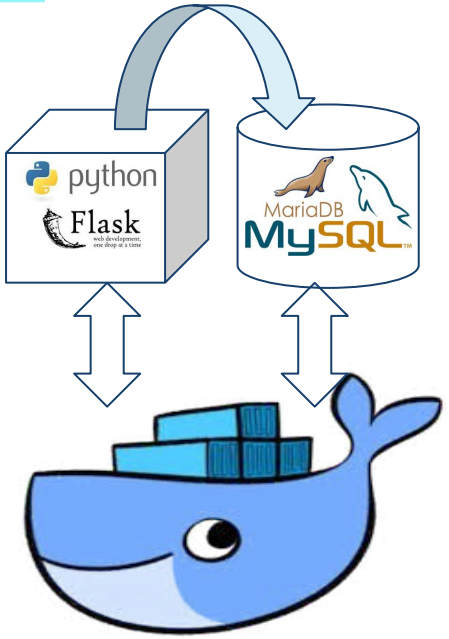
## 3.3.4 Docker - Compose

**What do you need and how to do it… ?**

Accessing the application:

1. You can use mysql Workbench or any MySQL dbms tool to access the database.
2. Use a browser to check the flask application web access.
3. Use Postman or any REST compliant tool to check flask app REST endpoints.



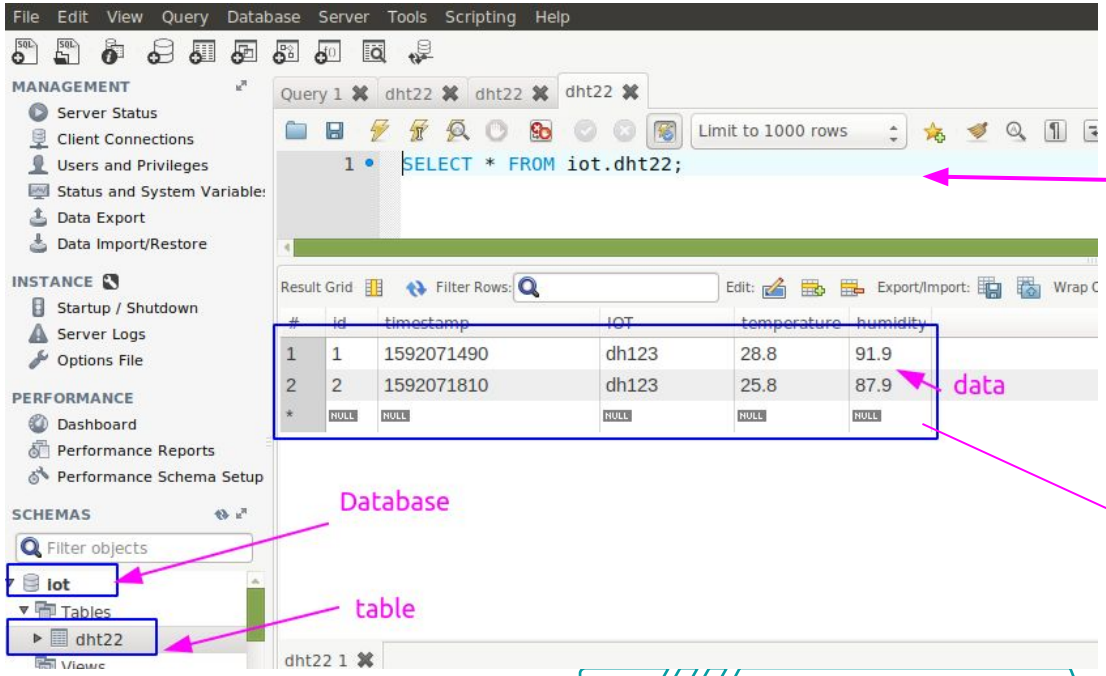

sending JSON data from Postman REST Client

## 3.3.4 Docker - Compose

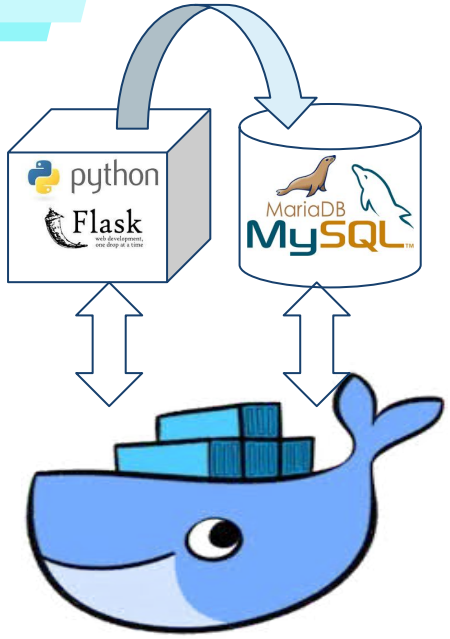
**What do you need and how to do it… ?**

Accessing the application:

1. You can use mysql Workbench or any MySQL dbms tool to access the database.
2. Use a browser to check the flask application web access.
3. Use Postman or any REST compliant tool to check flask app REST endpoints.




MySQL Workbench
RDBMS client

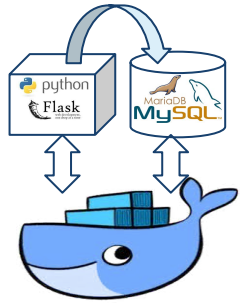



dht22 data with timestamp

| # | id | timestamp | IOT | temperature | humidity |
|---|---|---|---|---|---|
| 1 | 1 | 1592071490 | dht123 | 28.8 | 91.9 |
| 2 | 2 | 1592071810 | dht123 | 25.8 | 87.9 |
| 3 | 3 | 1592072049 | dht123 | 27.8 | 83.9 |

# 3.3 Virtualization and Cloud Platform



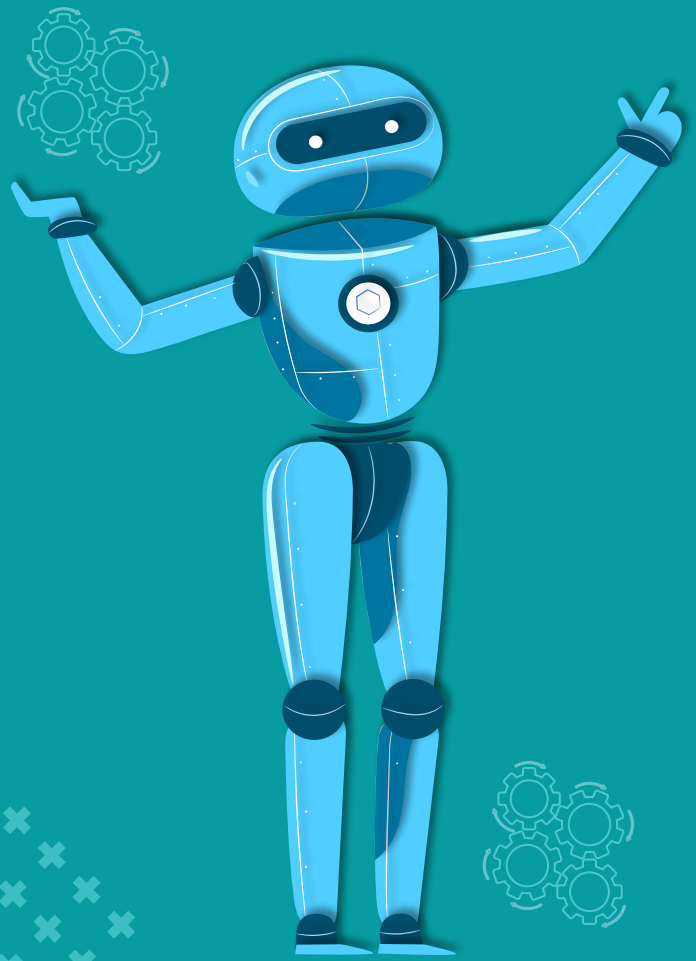

| | |
|---|---|
| **SaaS** | Google Apps, Dropbox, Salesforce, Cisco WebEx, Concur, GoToMeeting |
| **PaaS** | AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos, OpenShift |
| **IaaS** | DigitalOcean, Linode, Rackspace, Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure, Google Compute Engine (GCE) |

In conclusion, you can take advantage of both cloud and virtualization technologies to build powerful/scalable, flexible, reliable and easy to manage as your IoT infrastructure that is modern and accessible from your own LAN as well as anywhere in the world.

But you need to build a very good private / enterprise network to ensure stable data communication among IoTs, IoT gateways, computers / servers and internet equipment.

Please take note, that there are free services on the internet that can provide IoT infrastructure to host your IoT application, but their usage are limited to testing and development and personal use only. Example are Adafruits, Thingsboard, ThingSpeak and etc. There are also commercial platforms in the cloud but, you need to consider future growth and information security issue.

3.4



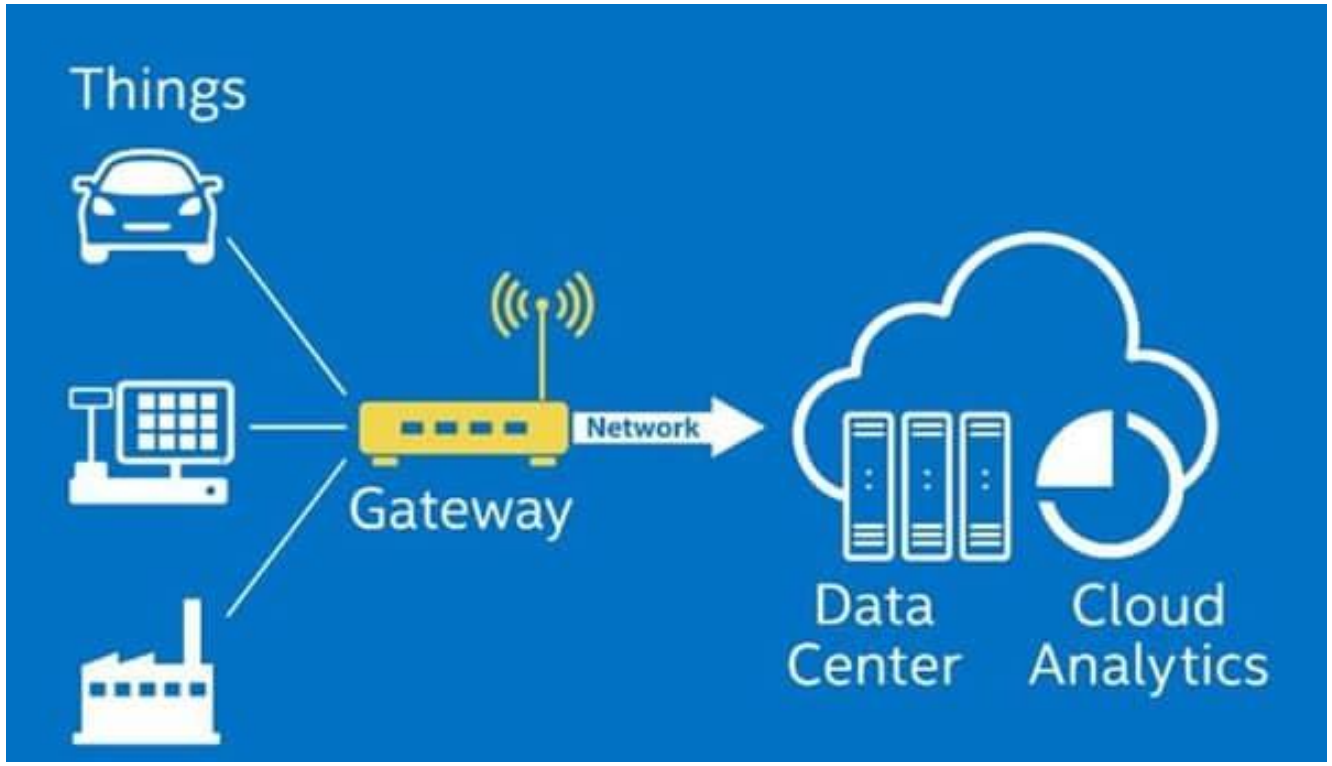
# IoT Gateway

- IoT Gateway
- IoT Gateway Connection Protocols
- Database
- IT Security

WIP

# 3.4 IoT Gateway

## 3.4.1 IoT Gateway


WIP

# 3.4 IoT Gateway

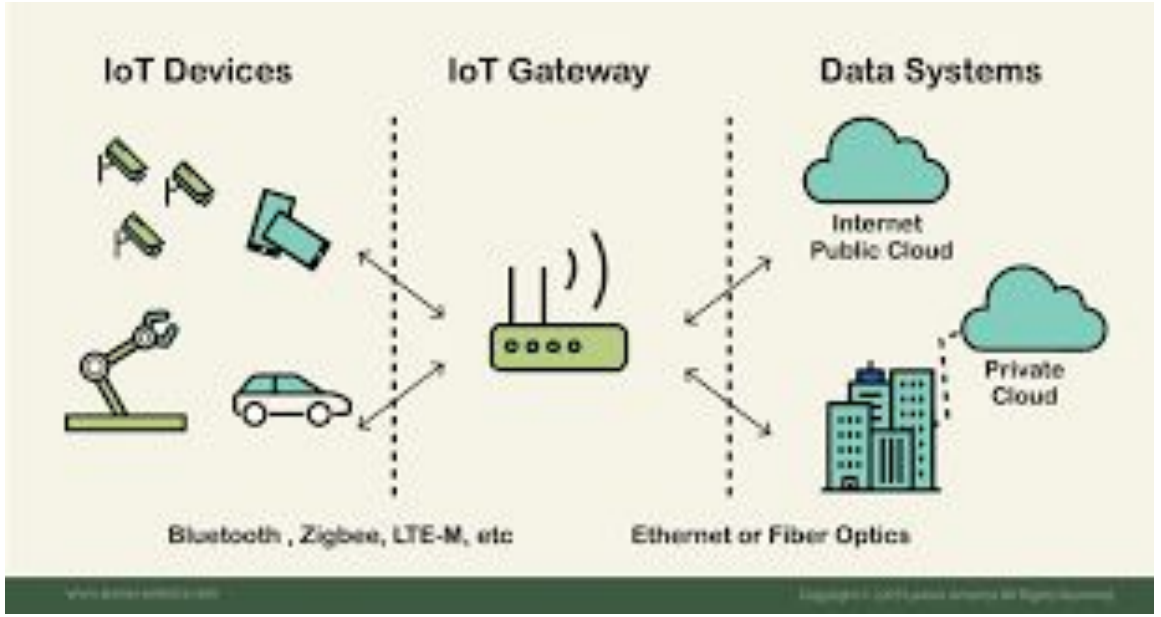
## 3.4.1 IoT Gateway

IoT gateways are essential components in IoT infrastructure.
It is because **they act as bridges between sensors/devices and other network devices and the cloud.**

Many sensors/devices will “talk” to a gateway and the gateway will then take all that information and “talk” to other devices and the cloud.

IoT gateway must support most common IoT communication technology such as WiFi, Ethernet, BLE, Serial RS232/485,  etc.

It must be able to communicate with IoT device and other IT equipment using communication protocols such as MQTT, OPC/UA, REST and popular database connection drivers.




Some advanced IoT gateways can also support data visualization via web interfaces and human-machine communication such as SMS and email.

WIP

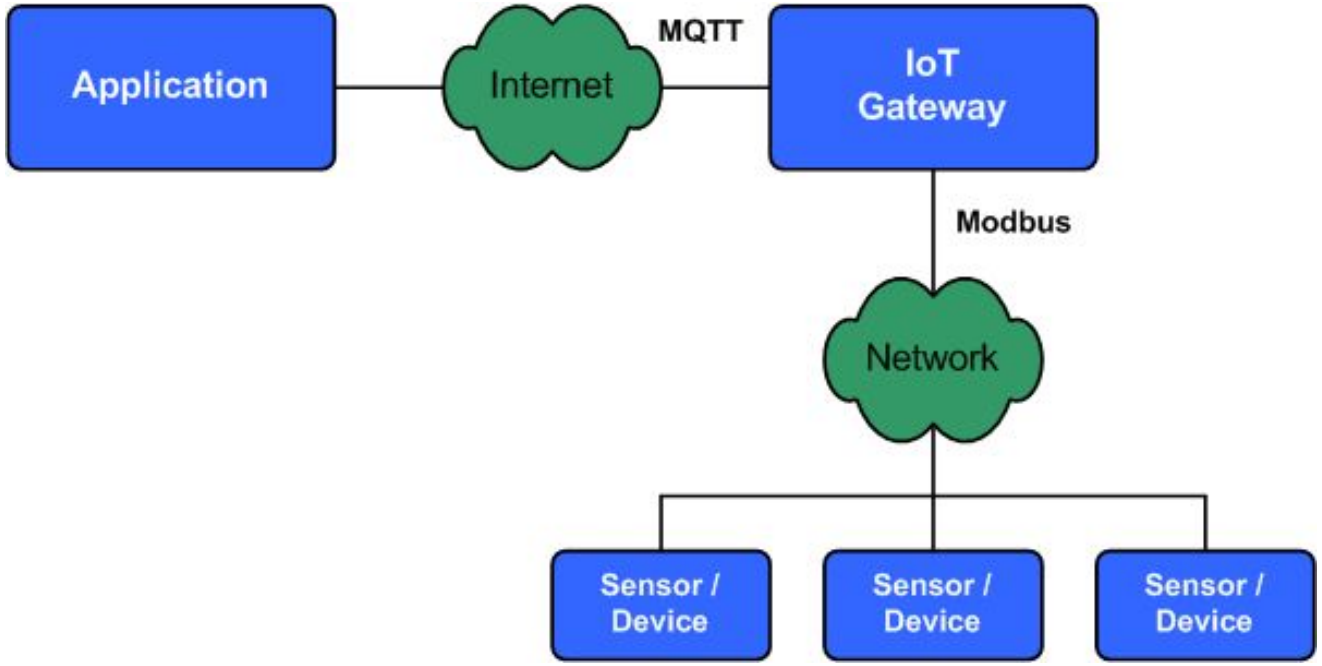
# 3.4 IoT Gateway

## 3.4.2 IoT Gateway Connection Protocols

IoT devices main functions are to do **sensors data digitization** and **controls some actuators** for automation.

In order to perform these, It must be able to communicate with other machines using IoT lightweight data communication protocols.

Most common IoT data communication protocols are as listed in the table (next slide)




WIP

# 3.4 IoT Gateway

## 3.4.2 IoT Gateway Connection Protocols

| | |
|---|---|
| MQTT (Message Queuing Telemetry Transport) | "The MQTT protocol enables a publish/subscribe messaging model in an extremely lightweight way.<br>It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium." |
| CoAP (Constrained Application Protocol) | CoAP is an application layer protocol that is intended for use in resource-constrained internet devices such as wireless sensor nodes.<br>CoAP is designed to easily translate to HTTP for simplified integration with the web, while also meeting specialized requirements such as multicast support, very low overhead, and simplicity. |
| REST (Representational state transfer) | RESTful HTTP is common web application data communication protocol that exchange data using compact JSON format over HTTP data transport. |
| TCP/IP TCP, UDP | Basic network socket data communication exchanging data packets.<br>Transmission Control Protocol (TCP) is a connection-oriented protocol that computers use to communicate over the internet.TCP provides error-checking and guarantees delivery of data and that packets will be delivered in the order they were sent<br>User Datagram Protocol (UDP) is a connectionless protocol that works just like TCP but assumes that error-checking and recovery services are not required. Instead, UDP continuously sends datagrams to the recipient whether they receive them or not. |

WIP

# 3.4 IoT Gateway

## 3.4.3 Database

IoT devices will generate a lot of data that need to be stored for visualization and analysis. We need database server to do this job and we have SQL and NoSQL database technologies available

1. SQL
    a. RDBMS

        i. RDBMS stands for Relational Database Management System.

        ii. RDBMS is the basis for SQL, and for all modern database systems such as MS SQL Server, IBM DB2, Oracle, MySQL, PostGres and Microsoft Access.

        iii. The data in RDBMS is stored in database objects called tables. A table is a collection of related data entries and it consists of columns and rows.

# 3.4 IoT Gateway

## 3.4.3 Database

IoT devices will generate a lot of data that need to be stored for visualization and analysis. We need database server to do this job and we have SQL and NoSQL database technologies available

2. NoSQL
    a. Key-Value Store Database
    b. Document Database
    c. Time Series Database
    d. Object Oriented Database
    e. Graph Database
    f. Multi Model Database

# 3.4 IoT Gateway

## 3.4.4 IT Security

Confidentiality

It's crucial in today's world for people to protect their sensitive, private information from unauthorized access.

Protecting confidentiality is dependent on being able to define and enforce certain access levels for information. In some cases, doing this involves separating information into various collections that are organized by who needs access to the information and how sensitive that information actually is - i.e. the amount of damage suffered if the confidentiality was breached.

Some of the most common means used to manage confidentiality include access control lists, volume and file encryption, and Unix file permissions.

# 3.4 IoT Gateway

## 3.4.4 IT Security

Integrity

Data integrity is what the "I" in CIA Triad stands for. This is an essential component of the CIA Triad and designed to protect data from deletion or modification from any unauthorized party, and it ensures that when an authorized person makes a change that should not have been made the damage can be reversed.

# 3.4 IoT Gateway

## 3.4.4 IT Security

Availability

This is the final component of the CIA Triad and refers to the actual availability of your data. Authentication mechanisms, access channels and systems all have to work properly for the information they protect and ensure it's available when it is needed.
High availability systems are the computing resources that have architectures that are specifically designed to improve availability. Based on the specific HA system design, this may target hardware failures, upgrades or power outages to help improve availability, or it may manage several network connections to route around various network outages.