UNIT 1: IOT INTRODUCTION

1.What Is IoT ?

- The Internet of Things (IoT) describes the network of physical objects—“things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet.
- **Basic goal of IoT :** “connect the unconnected.”

This means that objects not currently joined to the Internet will be connected so that they can communicate and interact with people and other objects.

2. Explain the impact of IoT ?

❖ **Internet of Things Positive Impacts:**

- Effective communication and Instant messaging services
- Increase business interactions, save vital time
- Less complicated banking, transactions, and shopping
- Access the latest news from anywhere in the world
- Run online course on virtual assistants using internet
- Professionals have IoT devices in healthcare, e-Commerce, and AI to help in doing jobs easily

❖ **Internet of Things Negative Impacts:**

- Easy availability of age unsuitable content materials and information
- Social networks disturb life personally and professionally
- Stealing data or hacking into IoT devices is easy
- Using the internet to spread bad scenarios via the IoT device is easy

3. Explain the drivers behind the network architecture ?

- To implement any networking concepts designing and understanding the network architecture is of utmost importance.
- The difference between IT and IoT networks is much like the difference between residential architecture and stadium architecture.
- The key difference between IT and IoT is the data.
- While IT systems are mostly concerned with reliable and continuous support of business applications such as email, web, databases and so on.
- IoT is all about the data generated by sensors and how that data is used.
- The essence of IoT architectures thus involves how the data is transported, collected, analyzed, and ultimately acted upon.
- **IOT ARCHITECTURAL DRIVERS :**
  - Scale
  - Security
  - Constrained Devices and Networks
  - Data
  - Legacy Device Support

1. **Scale:**
   - The scale of a typical IT network is on the order of several thousand devices—typically printers, mobile wireless devices, laptops, servers, and so on.
   - But when a scale of a network goes from a few thousand endpoints to a few millions the IT engineers lack a required skills to design a network that is intended to support millions of routable IP endpoints.
2. **Security:**
   - It has often been said that if World War III breaks out, it will be fought in cyberspace.
   - The frequency and impact of cyber attacks in recent years has increased dramatically.
   - Protecting corporate data from intrusion and theft is one of the mainfunctions of the IT department.
   - IT departments go to great lengths to protect servers, applications, and the network, setting up

defense-in-depth models with layers of security designed to protect the cyber crown jewels of the corporation.

- However, despite all the efforts mustered to protect networks and data, hackers still find ways to penetrate trusted networks.

3. **Constrained Devices and Networks**

- Most IoT sensors are designed for a single job, and they are typically small and inexpensive.
- This means they often have limited power, CPU, and memory, and they transmit only when there is something important.
- If an IT network has performance constraints, the solution is simple: Upgrade to a faster network.

4. **Data :**

- IoT devices generate a mountain of data.
- In general, most IT shops don't really care much about the unstructured chatty data generated by devices on the network.
- However, in IoT the data is like gold, as it is what enables businesses to deliver new IoT services that enhance the customer experience, reduce cost, and deliver new revenue opportunities2
- Although most IoT-generated data is unstructured, the insights it provides through analytics can revolutionize processes and create new business models.
- For ex : Imagine a smart city with a few hundred thousand smart streetlights, all connected through an IoT network.
- However, when all this data is combined, it can become difficult to manage and analyze effectively.

5. Legacy Device Support :

Supporting legacy devices in an IT organization is not usually a big problem.

If someone's computer or operating system is outdated, she simply upgrades.

If someone is using a mobile device with an outdated Wi-Fi standard, such as 802.11b or 802.11g, we can simply deny him access to the wireless network, and he will be forced to upgrade.

In OT systems, end devices are likely to be on the network for a very long time.

5. **Explain in brief IoTWF standardized architecture ?**

- A seven-layer IoT architectural reference model published by IoTWF architectural committee (Cisco, IBM, Rockwell Automation)
  - Edge computing
  - Data storage
  - Access

- Using this reference model, we are able to

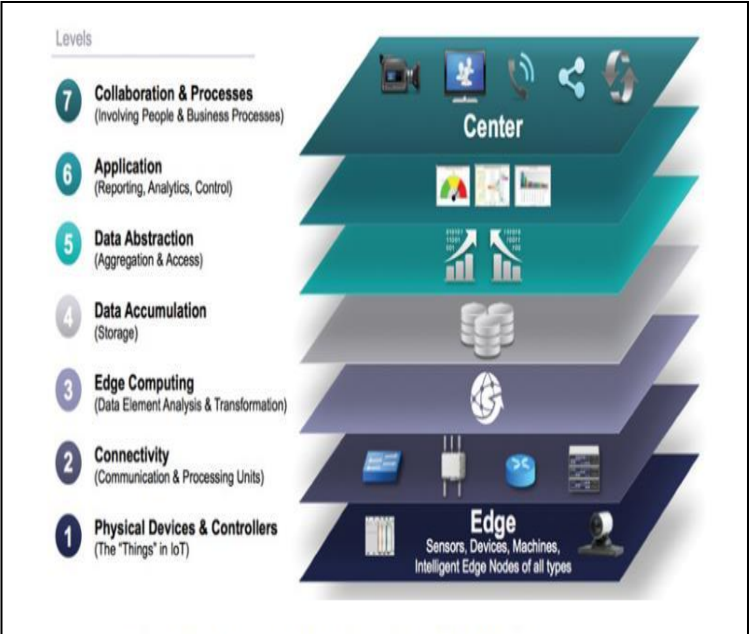

Levels
7 Collaboration & Processes (Involving People & Business Processes)
6 Application (Reporting, Analytics, Control)
5 Data Abstraction (Aggregation & Access)
4 Data Accumulation (Storage)
3 Edge Computing (Data Element Analysis & Transformation)
2 Connectivity (Communication & Processing Units)
1 Physical Devices & Controllers (The "Things" in IoT)
Center
Edge
Sensors, Devices, Machines, Intelligent Edge Nodes of all types

Figure 2-2 *IoT Reference Model Published by the IoT World Forum*

achieve the following:

- Decompose the IoT problem into smaller parts
- Identify different technologies at each layer and how they relate to one another
- Define a system in which different parts can be provided by different vendors
- Have a process of defining interfaces that leads to interoperability
- Define a tiered security model that is enforced at the transition points between levels

<u>Layer 1: Physical Devices and Controllers Layer</u>

- The various endpoint devices and sensors that send and receive information
- The size of these "things" can range from almost microscopic sensors to giant machines in a factory.
- Their primary function is generating data and being capable of being queried and/or controlled over a network.

<u>Layer 2: Connectivity Layer</u>

- The most important function of this IoT layer is the reliable and timely transmission of data.

Layer 2 function:

1. Communication between layer 1 devices.
2. Reliable delivery of information across the network
3. Switching and routing
4. Translation between protocol
5. Network level security

<u>Layer 3: Edge Computing Layer</u>

- Fog layer
- At this layer, the data reduction and converting network data flows into information that is ready for storage and processing by higher layers.
- One of the basic principles of this reference model is that information processing is initiated as early and as close to the edge of the network as possible .
- Another important function that occurs at Layer 3 is the evaluation of data to see if it can be filtered or aggregated before being sent to a higher layer.
- This also allows for data to be reformatted or decoded, making additional processing by other systems easier

Layer 3 fuction:

1. Evaluate and reformat data for processing at high level.
2. Filter data to reduce traffic higher level processing
3. Access data for alerting,Notification or other action.

<u>Layer 4: Data Accumulation Layer</u>

- Convert event based data to query based data.

<u>Layer 5: Data abstraction layer</u>

- Multiple data format and sementic from various source and confirms that the data set is complete and store multiple data using virtualiazation.

<u>Layer 6:Application Layer</u>

- Application may monitor,control,provide report based on the analysis of data.

<u>Layer 7:Collaboration and Process layer</u>

- Consumes and share the application information
- This layer can change business process and deliver the benifit of iot .
- Upper layer deals with handling and processing the IOT data generated by the bottom layer.

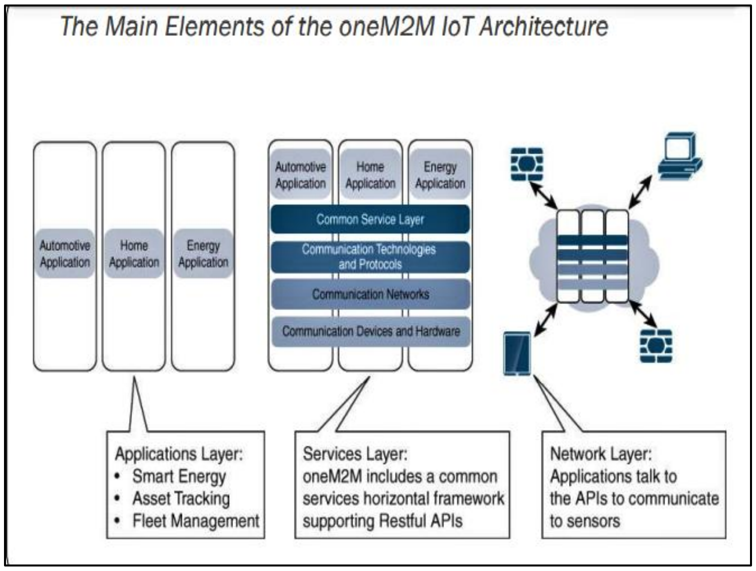
4. Explain in brief one M2M IoT standardized architecture ?

- In an effort to standardize the rapidly growing field of machine-to-machine (M2M) communications, the European Telecommunications Standards Institute (ETSI) created the M2M Technical Committee in 2008.
- The goal of this committee was to create a common architecture that would help accelerate the adoption of M2M applications and devices.
- oneM2M's framework focuses on IoT services, applications, and platforms
- One of the greatest challenges of IoT architecture is dealing with the heterogeneity of devices, software, and access methods.
- By developing a horizontal platform architecture, oneM2M is developing standards that allow interoperability at all levels of the IoT stack.

- This is where the oneM2M common services architecture comes in.

Three major domains:

- The application layer,
- The services layer,
- The network layer


The Main Elements of the oneM2M IoT Architecture
Automotive Application
Home Application
Energy Application
Common Service Layer
Communication Technologies and Protocols
Communication Networks
Communication Devices and Hardware
Applications Layer:
• Smart Energy
• Asset Tracking
• Fleet Management
Services Layer:
oneM2M includes a common services horizontal framework supporting Restful APIs
Network Layer:
Applications talk to the APIs to communicate to sensors

Application layer :

- The oneM2M architecture gives major attention to connectivity between devices and their applications.
- It includes the application-layer protocols and attempts to standardize northbound API
- definitions for interaction with business intelligence (BI) systems.

Services layer

- This layer is shown as a horizontal framework across the vertical industry applications
- At this layer, horizontal modules include the physical network that the IoT applications run on, the underlying management protocols, and the hardware.
- Examples include backhaul communications via cellular, MPLS networks, VPNs, and so on.
- Riding on top is the common services layer.
- This conceptual layer adds APIs and middleware supporting third-party services and applications

Network layer

- This is the communication domain for the IoT devices and endpoints.
- It includes the devices themselves and the communications network that links them
- Communications infrastructure include wireless mesh technologies, such as IEEE 802.15.4, and wireless point-to-multipoint systems, such as IEEE 801.11ah

5. Explain in detail the IoT Data Management and Compute Stack.

- The data generated by IoT sensors is one of the single biggest challenges in building an IoT system
- In the case of modern IT networks, the data sourced by a computer or server is typically generated by the client/server communications model, and it serves the needs of the application
- In sensor networks, the vast majority of data generated is unstructured and of very little use on its own
- In most cases, the processing location is outside the smart object.
- A natural location for this processing activity is the cloud.
- Smart objects need to connect to the cloud, and data processing is centralized.
- One advantage of this model is simplicity.
- Objects just need to connect to a central cloud application.
- That application has visibility over all the IoT nodes and can process all the analytics needed.
- This model also has limitations.

Minimizing latency:

- Milliseconds matter for many types of industrial systems
- Analyzing data close to the device that collected the data can make a difference

Conserving network bandwidth:

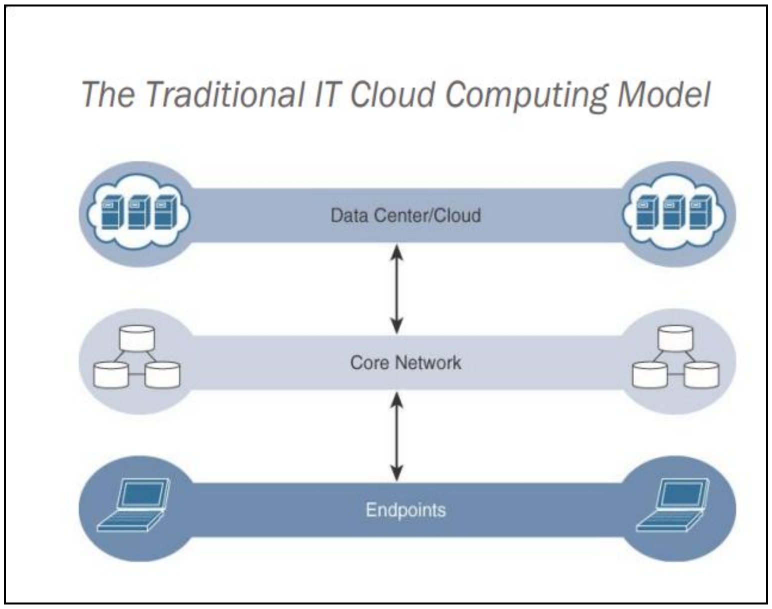
- Offshore oil rigs generate 500 GB of data weekly.
- Commercial jets generate 10 TB for every 30 minutes of flight.
- It is not practical to transport vast amounts of data from thousands or hundreds of thousands of edge devices to the cloud

Increasing local efficiency:

- Collecting and securing data across a wide geographic area with different environmental conditions may not be useful.

- Collecting and securing data across a wide geographic area with
- different environmental conditions may not be useful.
- Analyzing both areas in the same cloud system may not be necessary for immediate efficiency.

- IoT systems function differently.
- Several data-related problems need to be addressed:
- Bandwidth in last-mile IoT networks is very limited.
- When dealing with thousands/millions of devices, available bandwidth may be on order of tens of Kbps per device or even less.
- Latency can be very high.
- Big data is getting bigger. The concept of storing and analyzing all sensor data in the cloud is impractical



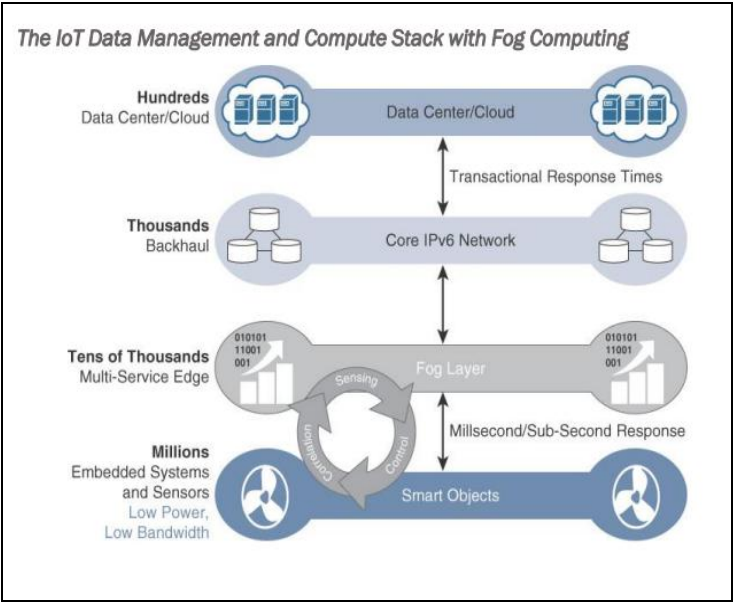
Fog Computing:

- Distribute data management throughout the IoT system, as close to the edge of the IP network as possible.
- The best-known embodiment of edge services in IoT is fog computing.
- Any device with computing, storage, and network connectivity can be a fog node.
- Examples include industrial controllers, switches, routers, embedded servers, and IoT gateways.
- Analyzing IoT data close to where it is collected minimizes latency, offloads gigabytes of network traffic from the core network, and keeps sensitive data inside

.

The defining characteristic of fog computing are as follows:

- Contextual location awareness and low latency
- Geographic distribution
- Deployment near IoT endpoints
- Wireless communication between the fog and the IoT endpoint
- Use for real-time interactions

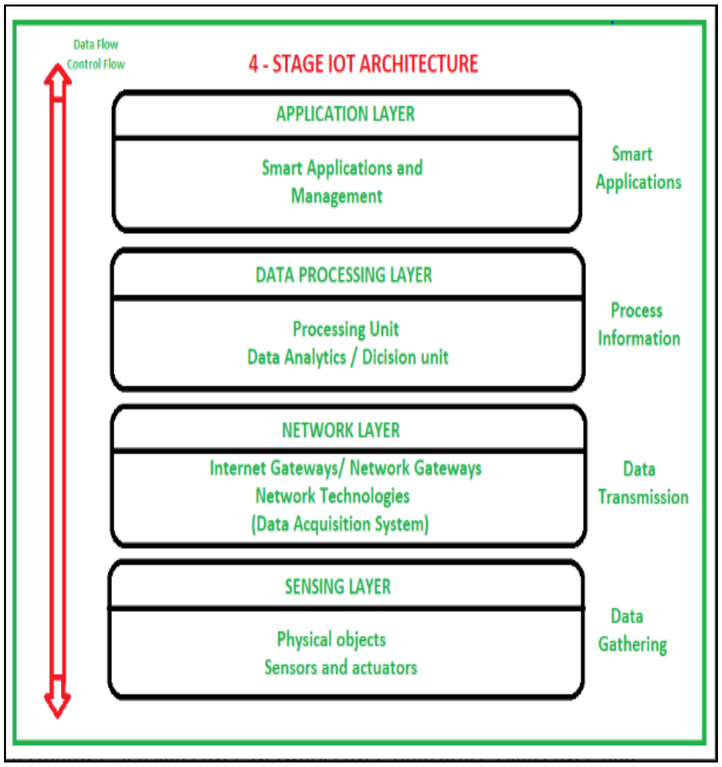


6 .Explain in brief simplified IoT architecture ?

- Internet of Things (IoT) technology has a wide variety of applications and use of Internet of Things is growing so faster.
- Depending upon different application areas of Internet of Things, it works accordingly as per it has been designed/developed.
- But it has not a standard defined architecture of working which is strictly followed universally.
- The architecture of IoT depends upon its functionality and implementation in different sectors. Still, there is a basic process flow based on which IoT is built.

- The whole variety of factors affecting IoT architecture, it's easier and more effective to find a reliable provider of IoT solutions.
- This decision will significantly reduce the number of resources spent on the way.

There are four layers of IOT architecture:



1. Application layer
2. Data processing layer
3. Network layer
4. Sensing layer

1.Sensing Layer –

- Sensors, actuators, devices are present in this Sensing layer.
- These Sensors or Actuators accepts data(physical/environmental parameters), processes data and emits data over network.

2.Network Layer –

- Internet/Network gateways, Data Acquisition System (DAS) are present in this layer.
- DAS performs data aggregation and conversion function (Collecting data and aggregating data then converting analog data of sensors to digital data etc).
- Advanced gateways which mainly opens up connection between Sensor networks and Internet also performs many basic gateway functionalities like malware protection, and filtering also some times decision making based on inputted data and data management services, etc.

3.Data processing Layer –

- This is processing unit of IoT ecosystem.
- Here data is analyzed and pre-processed before sending it to data center from where data is accessed by software applications often termed as business applications where data is monitored and managed and further actions are also prepared.
- So here Edge IT or edge analytics comes into picture.

4.Application Layer –

- This is last layer of 4 stages of IoT architecture.
- Data centers or cloud is management stage of data where data is managed and is used by end-user applications like agriculture, health care, aerospace, farming, defense, etc.

7.Challenges in Internet of things (IoT)?

- The Internet of Things (IoT) has fast grown to be a large part of how human beings live, communicate and do business.
- All across the world, web-enabled devices are turning our global rights into a greater switched-on area to live in.

Security challenges in IoT :

1. **Lack of encryption –**
    - Although encryption is a great way to prevent hackers from accessing data, it is also one of the leading IoT security challenges.
    - The result is an increase in attacks where hackers can easily manipulate the algorithms that were designed for protection.
2. **Insufficient testing and updating –**
    - With the increase in the number of IoT(internet of things) devices, IoT

manufacturers are more eager to produce and deliver their device as fast as they can without giving security too much of although.

- Most of these devices and IoT products do not get enough testing and updates and are prone to hackers and other security issues.

Design challenge in IoT :

1. **Battery life is a limitation –**
   - Issues in packaging and integration of small-sized chip with low weight and less power consumption.
   - If you've been following the mobile space, you've likely see how every yr it looks like there's no restriction in terms of display screen size.
2. **Increased cost and time to market –**
   - Embedded systems are lightly constrained by cost.
   - The need originates to drive better approaches when designing the IoT devices in order to handle the cost modelling or cost optimally with digital electronic components.
3. **Security of the system –**
   - Systems have to be designed and implemented to be robust and reliable and have to be secure with cryptographic algorithms and security procedures.
   - It involves different approaches to secure all the components of embedded systems from prototype to deployment.

Deployment challenges in IoT :

1. **Connectivity –**
   - It is the foremost concern while connecting devices, applications and cloud platforms.
   - Connected devices that provide useful front and information are extremely valuable.
   - But poor connectivity becomes a challenge where IoT sensors are required to monitor process data and supply information.
2. **Cross platform capability –**
   - IoT applications must be developed, keeping in mind the technological changes of the future.
   - Its development requires a balance of hardware and software functions.
   - It is a challenge for IoT application developers to ensure that the device and IoT platform drivers the best performance despite heavy device rates and fixings.
3. **Data collection and processing –**
   - In IoT development, data plays an important role. What is more critical here is the processing or usefulness of stored data.
   - Along with security and privacy, development teams need to ensure that they plan well for the way data is collected, stored or processed within an environment.

8. Explain the difference between Services layer and the Network layer for IoT architecture ?

Services layer:

- The horizontal modules include the physical network that the IoT applications run on, the underlying management protocols, and the hardware. Examples include backhaul communications via cellular, MPLS networks, VPNs, and so on.
- This conceptual layer adds APIs and middleware supporting third-party services and applications.
- One of the stated goals of oneM2M is to "develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software nodes.

Network layer:

- It includes the devices and the communications infrastructure which include wireless mesh technologies, such as IEEE

802.15.4, and wireless point-to-multipoint systems, such as IEEE 801.11ah.

- Also included are wired device connections, such as IEEE 1901 power line communications.
- The device domain includes the gateway device, which provides communications up into the core network and acts as a demarcation point between the device and network domains.

UNIT 2: SMART OBJECT

1. Write a short note on smart objects.

- The concept of smart in IoT is used for physical objects that are active, digital, networked, can operate to some extent autonomously, reconfigurable and has local control of the resources.
- The smart objects need energy, data storage, etc.
- A **smart object** is an object that enhances the interaction with other smart objects as well as with people also.
- The world of IoT is the network of interconnected heterogeneous objects (such as smart devices, smart objects, sensors, actuators, RFID, embedded computers, etc.) uniquely addressable and based on standard communication protocols.
- Smart objects are utilized widely to transform the physical environment around us to a digital world using the Internet of things (IoT) technologies.
- A smart object carries blocks of application logic that make sense for their local situation and interact with human users.

2. Explain different types of sensor?

- sensors are used in the architecture of IOT devices.
- **Sensors** are used for sensing things and devices etc.
- A device that provides a usable output in response to a specified measurement.
- The output of the sensor is a signal which is converted to a human-readable form like changes in characteristics, changes in resistance, capacitance, impedance etc.
- The sensor attains a physical parameter and converts it into a signal suitable for processing (e.g. electrical, mechanical, optical) the characteristics of any device or material to detect the presence of a particular physical quantity.

Categories of sensor:

- **Active or passive:**
  Sensors can be categorized based on whether they produce an energy output and typically require an external power supply (active) or whether they simply receive energy and typically require no external power supply (passive).
- **Invasive or non-invasive**:
  Sensors can be categorized based on whether a sensor is part of the environment it is measuring (invasive) or external to it (non-invasive).
- **Contact or no-contact**:
  Sensors can be categorized based on whether they require physical contact with what they are measuring (contact) or not (no-contact).
- **Absolute or relative:**
  Sensors can be categorized based on whether they measure on an absolute scale (absolute) or based on a difference with a fixed or variable reference value (relative).
- **Area of application:**
  Sensors can be categorized based on the specific industry or vertical where they are being used.
- **How sensors measure:**
  Sensors can be categorized based on the physical mechanism used to measure sensory input (for example, thermoelectric,

electrochemical, piezoresistive, optic, electric, fluid mechanic, photoelastic).

- **What sensors measure**:
  Sensors can be categorized based on their applications or what physical variables they measure.

Types of sensor:

Temperature Sensors

- In the past, IoT temperature sensors have been used for heat, ventilation, and air conditioning systems (HVAC), refrigerators, and other similar devices used for environmental control.
- However, the emergence of IoT has seen its role expand.
- Nowadays, you can find temperature sensors throughout industries such as manufacturing and agriculture.

Pressure Sensors

- An IoT pressure sensor is any device that senses pressure and converts it into an electric signal.
- The level of voltage given out by the sensor depends on the level of pressure applied.
- These sensors enable IoT systems that monitor systems and devices that are pressure propelled.
- If there's any deviation from standard pressure ranges, the device notifies the administrator of the problem.

Humidity Sensors

- IoT humidity sensors measure the amount of water vapor in the air.
- In scientific terms, they measure Relative Humidity (RH).
- This kind of sensor is usually used in addition to IoT temperature sensors when a manufacturing process requires absolute perfect working conditions.
- They're usually found in heating, ventilation, and air conditioning (HVAC) systems – in both the home and business settings.

Image Sensors

- IoT image sensors are used to convert images into electronic signals.
- These are then either displayed or become electronically stored files.
- The most common use of image sensors is in digital cameras and IoT WiFi modules.

Level Sensors

- Level sensors are used to detect the levels of certain types of objects.

- These include liquids, granular materials, and powders.
- As you can imagine, this kind of sensor is useful and used in many different industries and applications.
- These include:
  - Oil manufacturing
  - Beverage manufacturing
  - Food manufacturing
  - Water treatment

Gas sensor:

- Gas sensors monitor and detect changes in the air.
- These sensors are vital to our safety as they're able to detect the presence of potentially harmful or even toxic gases.
- Gas sensors are most commonly used within the mining, oil and gas, and chemical research.

Motion Detector Sensors

- Not to be confused with proximity sensors, motion detector sensors are used to detect physical movement in a given area.
- In turn, this then sets off an electronic signal.
- The most obvious use of this is within the security industry.
- Businesses use motion detector sensors in areas where there should be no movement.
- These IoT sensors can also be found in many devices within modern commercial buildings

3. **Explain the characteristics of smart objects.**

Processing unit:

- A smart object has some type of processing unit for acquiring data, processing and analyzing sensing information received by the sensor(s), coordinating control signals to any actuators, and controlling a variety of functions on the smart object, including the communication and power systems.

Sensor(s) and/or actuator(s):

- A smart object is capable of interacting with the physical world through sensors and actuators.
- As described in the previous sections, a sensor learns and measures its environment, whereas an actuator is able to produce some change in the physical world.

Communication device:

- The communication unit is responsible for connecting a smart object with other smart objects and the outside world (via the network).
- Power source: Smart objects have components that need to be powered.
- Interestingly, the most significant power consumption usually comes from the communication unit of a smart object.

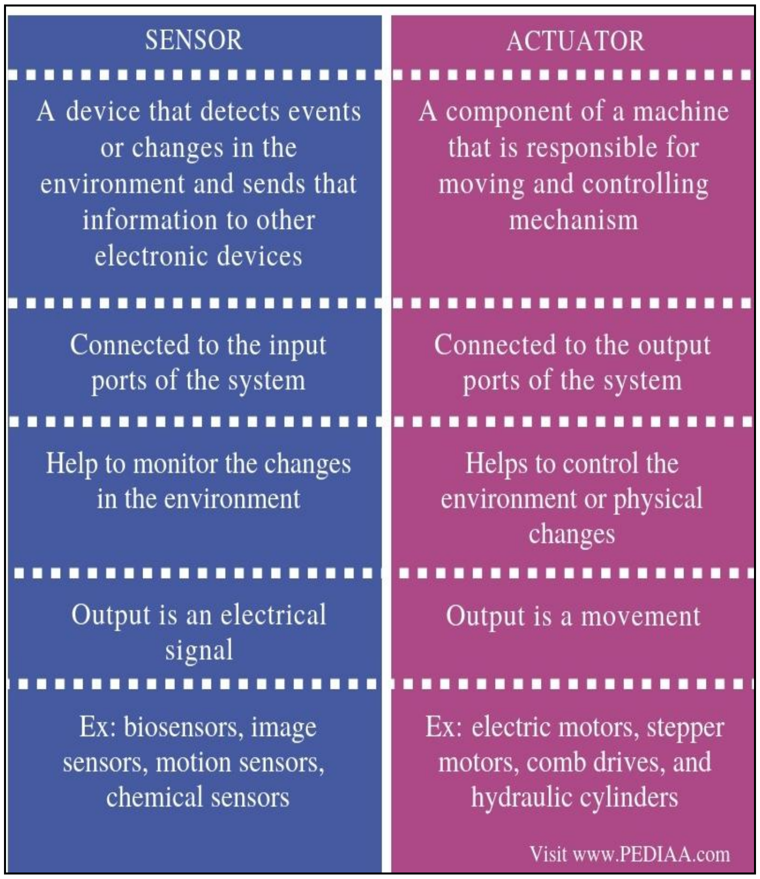
Power source:

- Smart objects have components that need to be powered.
- Interestingly, the most significant power consumption usually comes from the communication unit of a smart object.
- As with the other three smart object building blocks, the power requirements also vary greatly from application to application.
- Typically, smart objects are limited in power, are deployed for a very long time, and are not easily accessible.
- This combination, especially when the smart object relies on battery power, implies that power efficiency, judicious power management, sleep modes, ultra-low power consumption hardware, and so on are critical design elements.

5. Write a short note on wireless sensor networks.

- Wireless sensor networks are made up of wirelessly connected smart objects, which are sometimes referred to as motes.
- The fact that there is no infrastructure to consider with WSNs is surely a powerful advantage for flexible deployments, but there are a variety of design constraints to consider with these wirelessly connected smart objects.
- limitations of the smart objects in WSNs:
  - Limited processing power
  - Limited memory
  - Lossy communication
  - Limited transmission speeds
  - Limited power
- Smart objects with limited processing, memory, power, and so on are often referred to as constrained nodes.
- These limitations greatly influence how WSNs are designed, deployed, and utilized.
  - The fact that individual sensor nodes are typically so limited is a reason that they are often deployed in very large numbers.
  - As the cost of sensor nodes continues to decline, the ability to deploy highly redundant sensors becomes increasingly feasible.
  - Because many sensors are very inexpensive and correspondingly inaccurate, the ability to deploy smart objects redundantly allows for increased accuracy
  - Such large numbers of sensors permit the introduction of hierarchies of smart objects.

6. **Difference between sensor and actuator?**



SENSOR	ACTUATOR
A device that detects events or changes in the environment and sends that information to other electronic devices	A component of a machine that is responsible for moving and controlling mechanism
Connected to the input ports of the system	Connected to the output ports of the system
Help to monitor the changes in the environment	Helps to control the environment or physical changes
Output is an electrical signal	Output is a movement
Ex: biosensors, image sensors, motion sensors, chemical sensors	Ex: electric motors, stepper motors, comb drives, and hydraulic cylinders

Visit www.PEDIAA.com

7. **What are the different criteria for connecting smart objects?**

- IoT devices and sensors must be connected to the network for their data to be utilized.
- In addition to the wide range of sensors, actuators, and smart objects that make up IoT, there are also a number of different protocols used to connect them.

Communication criteria:

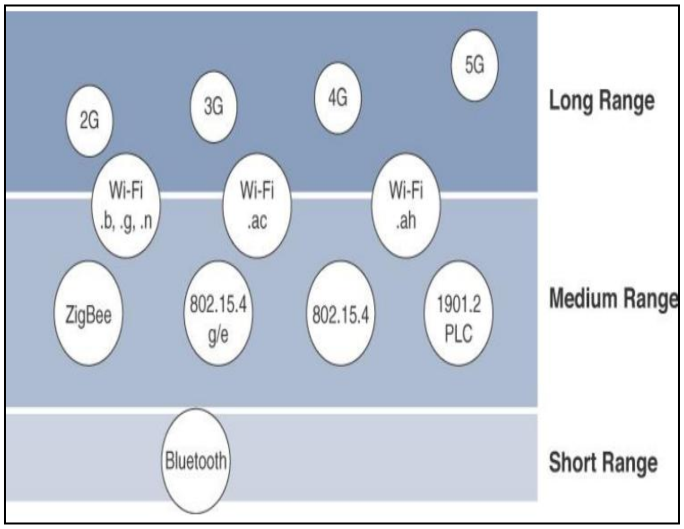
- Range
- Frequency Bands
- Power Consumption
- Topology
- Constrained Devices
- Constrained-Node Networks

Range :

- How far does the signal need to be propagated?
- That is, what will be the area of coverage for a selected wireless technology?
- Should indoor versus outdoor deployments be differentiated?

These are the questions asked when discussing wired and wireless access technologies.

- The simplest approach to answering these types of questions is to categorize these technologies as shown in Fig, breaking them down into the following range



1. **Short range:**

- The classical wired example is a serial cable.
- Wireless short-range technologies are often considered as an alternative to a serial cable, supporting tens of meters of maximum distance between two devices.
- Examples of short-range wireless technologies are IEEE 802.15.1 Bluetooth and IEEE 802.15.7 Visible Light Communications (VLC)
- These shortrange communication methods are found in only a minority of IoT installations. In some cases, they are not mature enough for production deployment.

2. **Medium range:**

- This range is the main category of IoT access technologies.
- In the range of tens to hundreds of meters, many specifications and implementations are available.
- The maximum distance is generally less than 1 mile between two devices

- Examples of medium-range wireless technologies include IEEE 802.11 Wi-Fi, IEEE 802.15.4, and 802.15.4g WPAN.
- Wired technologies such as IEEE 802.3 Ethernet and IEEE 1901.2 Narrowband Power Line Communications (PLC) may also be classified as medium range, depending on their physical media characteristics.

3. **Long range:**

- Distances greater than 1 mile between two devices require long-range technologies.
- Wireless examples are cellular (2G, 3G, 4G) and some applications of outdoor IEEE 802.11 Wi-Fi and Low-Power Wide-Area (LPWA) technologies.
- LPWA communications have the ability to communicate over a large area without consuming much power.
- These technologies are therefore ideal for battery-powered IoT sensors

<u>Frequency Bands :</u>

- Radio spectrum is regulated by countries and/or organizations, such as the International Telecommunication Union (ITU) and the Federal Communications Commission (FCC).
- These groups define the regulations and transmission requirements for various frequency bands .
- For example, portions of the spectrum are allocated to types of telecommunications such as radio, television, military, and so on.
- Focusing on IoT access technologies, the frequency bands leveraged by wireless communications are split between licensed and unlicensed bands.
- Licensed spectrum is generally applicable to IoT longrange access technologies .
- In order to utilize licensed spectrum, users must subscribe to services when connecting their IoT devices .
- The ITU has also defined unlicensed spectrum for the industrial, scientific, and medical (ISM) portions of the radio bands.
- These frequencies are used in many communications technologies for short-range devices (SRDs).
- Unlicensed means that no guarantees or protections are offered in the ISM bands for device communications
- ISM bands for IoT access
- 2.4 GHz band as used by IEEE 802.11b/g/n Wi-Fi IEEE 802.15.1 Bluetooth
- IEEE 802.15.4 WPAN
- Unlicensed spectrum is usually simpler to deploy than licensed because it does not require a service provider.
- The frequency of transmission directly impacts how a signal propagates and its practical maximum range.

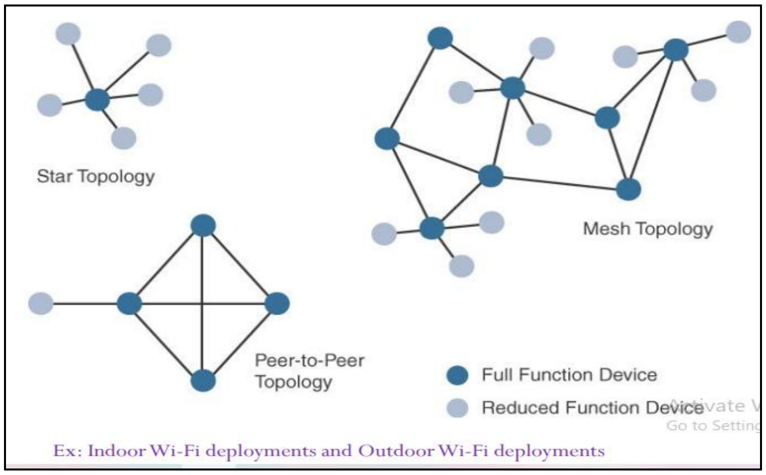
<u>Power Consumption</u>

- Powered nodes and battery-powered nodes
- A powered node has a direct connection to a power source, and communications are usually not limited by power consumption criteria.
- However, ease of deployment of powered nodes is limited by the availability of a power source, which makes mobility more complex
- Battery-powered nodes bring much more flexibility to IoT devices.
- These nodes are often classified by the required lifetimes of their batteries.
- A new wireless environment known as Low-Power WideArea (LPWA)
- Battery-powered nodes are often placed in a “sleep mode” to preserve battery life when not transmitting
- Wired IoT access technologies consisting of powered nodes are not exempt from power optimization

<u>Topology</u>

- Among the access technologies available for connecting IoT devices, three main topology schemes are dominant: star, mesh, and peer-to-peer.

- For long-range and short-range technologies, a star topology is prevalent.
- Star topologies utilize a single central base station or controller to allow communications with endpoints .
- For medium-range technologies, a star, peer-to-peer, or mesh topology is common.
- Peer-to-peer topologies allow any device to communicate with any other device as long as they are in range of each other.



Constrained device:

Class 0:

- Class 0 devices have constraints in memory(<<10KiB of RAM and <<100KiB of Flash) and processing capabilities.
- These devices has severe constraints to communicate securely with internet, so they typically pre-configured and are connected to proxies, gateways, or servers for internet communication.

Class 1:

- Class 1 devices can have low power IoT stack [UDP, CoAP, leight weigh security protocols like DTLS etc]
- But quite constrained in code space and processing capabilities to employing a full protocol stack such as using HTTP, TLS, and related security protocols and data representations with out a gateway.

Class 2:

- Class 2 devices are less constrained and can perform at par with mobiles phones/notebooks in supporting most the protocol stacks.
- They have to be lightweight with energy-efficient protocols and less bandwidth consumption.
- Using Class 2 devices might reduce development costs and increase the interoperability.

Constrained-Node Networks

- IEEE 802.15.4 and 802.15.4g RF, IEEE 1901.2a PLC, LPWA, and IEEE 802.11ah access technologies
- Constrained-node networks are often referred to as low-power and lossy networks (LLNs).
- Low power – battery powered constraints
- Lossy network -- network performance may suffer from interference and variability due to harsh radio environments
- Protocols that can be used for constrained-node networks must be evaluated in the context of the following characteristics:
- data rate and throughput, latency and determinism, and overhead and payload.

Data Rate and Throughput

- The data rates available from IoT access technologies range from 100 bps with protocols such as Sigfox to tens of megabits per second with technologies such as LTE and IEEE 802.11ac
- However, the actual throughput is less
- Technologies not particularly designed for IoT, such as cellular and Wi-Fi, match up well to IoT applications with high bandwidth requirements
- Short-range technologies can also provide medium to high data rates that have enough throughput to connect a few endpoints.
- For example, Bluetooth sensors that are now appearing on connected wearables fall into this category.
- In this case, the solutions focus more on footprint and battery lifetime than on data rate.
- The IoT access technologies developed for constrained nodes are optimized for low power consumption, but they are also limited in terms of data rate

- Another characteristic of IoT devices is that a majority of them initiate the communication.

Latency and Determinism

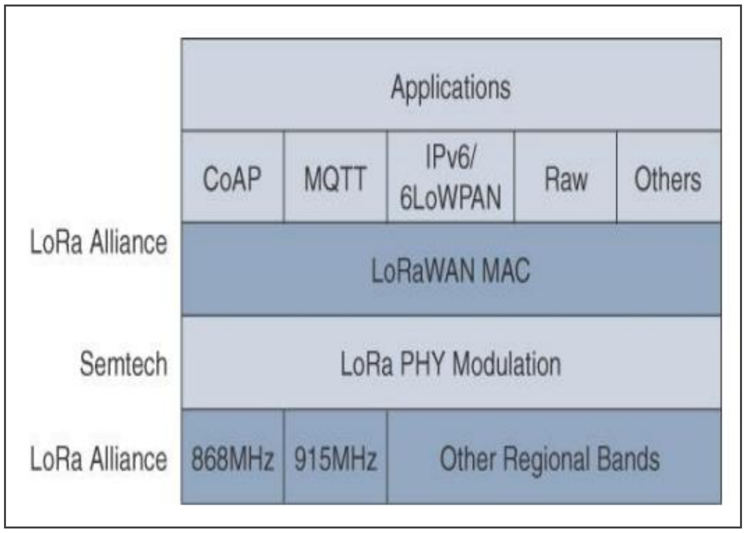
- Much like throughput requirements, latency expectations of IoT applications should be known when selecting an access technology.
- This is particularly true for wireless networks, where packet loss and retransmissions due to interference, collisions, and noise are normal behaviors.

Overhead and Payload .

- When considering constrained access network technologies, it is important to review the MAC payload size characteristics required by applications.
- You should be aware of any requirements for IP.
- The minimum IPv6 MTU size is expected to be 1280 bytes.

8. Explain the characteristics of LoraWAN.

- Low-Power Wide-Area (LPWA) adapted for long-range and battery powered endpoints
- LoRaWAN is unlicensed-band LPWA technology
- LoRa was a physical layer, or Layer 1, modulation that was developed by a French company named Cycleo
- Later, Cycleo was acquired by Semtech
- Optimized for long-range, two-way communications and low power consumptio



LoRaWAN Layers:

Physical Layer

- Semtech LoRa modulation is based on chirp spread spectrum modulation
- Chirp - Compressed High Intensity Radar Pulse
- Lower data rate and increase the communication distance Understanding LoRa gateways is critical to understanding a LoRaWAN system.
- A LoRa gateway is deployed as the center hub of a star network architecture.
- It uses multiple transceivers and channels and can demodulate multiple channels at once or even demodulate multiple signals on the same channel simultaneously.
- The data rate in LoRaWAN varies depending on the frequency bands and adaptive data rate (ADR).
- ADR is an algorithm that manages the data rate and radio signal for each endpoint.
- An important feature of LoRa is its ability to handle various data rates via the spreading factor.
- Devices with a low spreading factor (SF) achieve less distance in their communications but transmit at faster speeds, resulting in less airtime.
- A higher SF provides slower transmission rates but achieves a higher reliability at longer distances.

MAC Layer

- The MAC layer is defined in the LoRaWAN specification.
- This layer takes advantage of the LoRa physical layer and classifies LoRaWAN endpoints to optimize their battery life and ensure downstream communications to the LoRaWAN endpoints.
- The LoRaWAN specification documents three classes of LoRaWAN devices:

Class A:

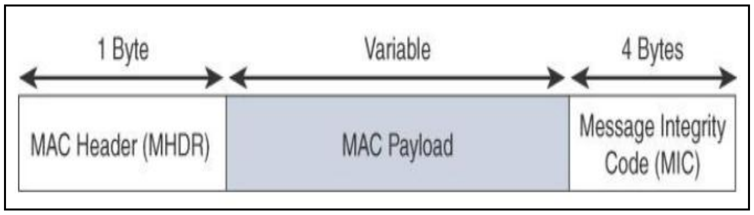
- This class is the default implementation. Optimized for battery powered nodes, it allows bidirectional communications, where a given node is able to receive downstream traffic after transmitting.
- Two receive windows are available after each transmission

Class B:

- This class was designated "experimental" in LoRaWAN 1.0.1 until it can be better defined.
- A Class B node or endpoint should get additional receive windows compared to Class A, but gateways must be synchronized through a beaconing process

Class C:

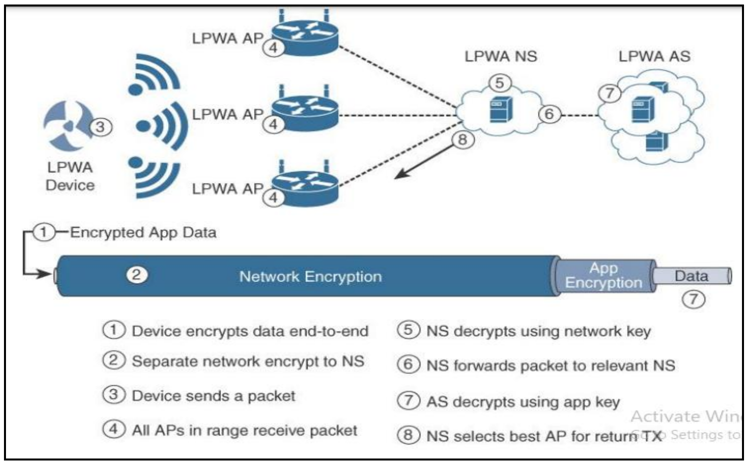
- This class is particularly adapted for powered nodes.
- This classification enables a node to be continuously listening by keeping its receive window open when not transmitting.
- LoRaWAN messages, either uplink or downlink, have a PHY payload composed of a 1-byte MAC header, a variable-byte MAC payload, and a MIC that is 4 bytes in length.
- The MAC payload size depends on the frequency band and the data rate, ranging from 59 to 230 bytes for the 863–870 MHz band and 19 to 250 bytes for the 902–928 MHz band.



- In version 1.0.x, LoRaWAN utilizes six MAC message types.
- LoRaWAN devices use join request and join accept messages for activation and joining the network.
- The other message types are unconfirmed data up/down and confirmed data up/down. A "confirmed" message is one that must be acknowledged, and "unconfirmed" signifies that the end device does not need to acknowledge.
- "up/down" is simply a directional notation identifying whether the message flows in the uplink or downlink path.
- Uplink messages are sent from endpoints to the network server and are relayed by one or more LoRaWAN gateways.
- Downlink messages flow from the network server to a single endpoint and are relayed by only a single gateway
- LoRaWAN endpoints are uniquely addressable through a variety of methods, including the following:
  - An endpoint can have a global end device ID or DevEUI represented as an IEEE EUI-64 address.
  - An endpoint can have a global application ID or AppEUI represented as an IEEE EUI-64 address that uniquely identifies the application provider, such as the owner, of the end device
  - In a LoRaWAN network, endpoints are also known by their end device address, known as a DevAddr, a 32-bit address.
  - The 7 most significant bits are the network identifier (NwkID), which identifies the LoRaWAN network. The 25 least significant bits are used as the network address (NwkAddr) to identify the endpoint in the network.

Security

- LoRaWAN endpoints must implement two layers of security, protecting communications and data privacy across the networ


LPWA AP
LPWA AP
LPWA AP
LPWA NS
LPWA AS
LPWA Device
1 Encrypted App Data
2 Network Encryption
App Encryption
Data
1 Device encrypts data end-to-end
2 Separate network encrypt to NS
3 Device sends a packet
4 All APs in range receive packet
5 NS decrypts using network key
6 NS forwards packet to relevant NS
7 AS decrypts using app key
8 NS selects best AP for return TX

- The first layer, called "network security" but applied at the MAC layer, guarantees the authentication of the endpoints by the LoRaWAN network server.
- Also, it protects LoRaWAN packets by performing encryption based on AES
- Each endpoint implements a network session key (NwkSKey), used by both itself and the LoRaWAN network server.
- This can be achieved through one of the two join mechanisms:

Activation by personalization (ABP):

- Endpoints don't need to run a join procedure as their individual details, including DevAddr and the NwkSKey and AppSKey session keys, are preconfigured and stored in the end device.
- This same information is registered in the LoRaWAN network server

Over-the-air activation (OTAA):

- Endpoints are allowed to dynamically join a particular LoRaWAN network after successfully going through a join procedure.
- The join procedure must be done every time a session context is renewed.
- During the join process, which involves the sending and receiving of MAC layer join request and join accept messages, the node establishes its credentials with a LoRaWAN network server, exchanging its globally unique DevEUI, AppEUI, and AppKey.
- The AppKey is then used to derive the session NwkSKey and AppSKey keys

9. **Explain the IEEE 802.15.4 technology in detail.**
    - IEEE 802.15.4 is a low-cost, low-data-rate wireless access technology for devices that are operated or work on batteries.
    - This describes how low-rate wireless personal area networks (LR-WPANs) function.

Properties:

1. **Standardization and alliances:**

- It specifies low-data-rate PHY and MAC layer requirements for wireless personal area networks(WPAN).
- IEEE 802.15. Protocol Stacks include:

ZigBee:

- ZigBee is a Personal Area Network task group with a low rate task group 4.
- It is a technology of home networking. ZigBee is a technological standard created for controlling and sensing the network.
- As we know that ZigBee is the Personal Area network of task group 4 so it is based on IEEE 802.15.4 and is created by Zigbee Alliance.

6LoWPAN:

- The 6LoWPAN system is used for a variety of applications including wireless sensor networks.
- This form of wireless sensor network sends data as packets and uses IPv6 – providing the basis for the name – IPv6 over Low power Wireless Personal Area Networks.

ZigBee IP:

- Zigbee is a standards-based wireless technology that was developed for low-cost and low-power wireless machine-to-machine (M2M) and internet of things (IoT) networks.

ISA100.11a:

- It is a mesh network that provides secure wireless communication to process control.

Wireless HART:

- It is also a wireless sensor network technology, that makes use of time-synchronized and self-organizing architecture.

Thread:

- Thread is an IPv6-based networking protocol for low-power Internet of Things devices in IEEE 802.15. 4-2006 wireless mesh network. Thread is independent.

2. **Physical Layer:**

- This standard enables a wide range of PHY options in ISM bands, ranging from 2.4 GHz to sub-GHz frequencies.

- IEEE 802.15.4 enables data transmission speeds of 20 kilobits per second, 40 kilobits per second, 100 kilobits per second, and 250 kilobits per second. The fundamental structure assumes a 10-meter range and a data rate of 250 kilobits per second.
- To further reduce power usage, even lower data rates are possible.
- IEEE 802.15.4 regulates the RF transceiver and channel selection, and even some energy and signal management features, at the physical layer.
- Based on the frequency range and data performance needed, there are now six PHYs specified.
- Four of them employ frequency hopping techniques known as Direct Sequence Spread Spectrum (DSSS).
- Both PHY data service and management service share a single packet structure so that they can maintain a common simple interface with MAC.

3. **MAC layer:**

- The MAC layer provides links to the PHY channel by determining that devices in the same region will share the assigned frequencies.
- The scheduling and routing of data packets are also managed at this layer.
- The 802.15.4 MAC layer is responsible for a number of functions like:
  - ❖ Beaconing for devices that operate as controllers in a network.
  - ❖ used to associate and dissociate PANs with the help of devices.
  - ❖ The safety of the device.
  - ❖ Consistent communication between two MAC devices that are in a peer-to-peer relationship.
- Several established frame types are used by the MAC layer to accomplish these functions. In 802.15.4, there are four different types of MAC frames:
  - ❖ frame of data
  - ❖ Frame for a beacon
  - ❖ Frame of acknowledgment
  - ❖ Frame for MAC command

4. **Topology:**

- Networks based on IEEE 802.15.4 can be developed in a star, peer-to-peer, or mesh topology. Mesh networks connect a large number of nodes.
- This enables nodes that would otherwise be out of range to interact with each other to use intermediate nodes to relay data.

5. **Security:**

- For data security, the IEEE 802.15.4 standard employs the Advanced Encryption Standard (AES) with a 128-bit key length as the basic encryption technique.
- Activating such security measures for 802.15.4 significantly alters the frame format and uses a few of the payloads.
- The very first phase in activating AES encryption is to use the Security Enabled field in the Frame Control part of the 802.15.4 header
- For safety, this field is a single bit which is assigned to 1.
- When this bit is set, by taking certain bytes from its Payload field, a field known as the Auxiliary Security Header is formed following the Source Address field.

6. **Competitive Technologies:**

- The IEEE 802.15.4 PHY and MAC layers serve as a basis for a variety of networking profiles that operate in different IoT access scenarios.
- DASH7 is a competing radio technology with distinct PHY and MAC layers.

Advantages of IEEE 802.15.4:

IEEE 802.15.4 has the following advantages:

- cheap cost
- long battery life,
- Quick installation
- simple
- extensible protocol stack

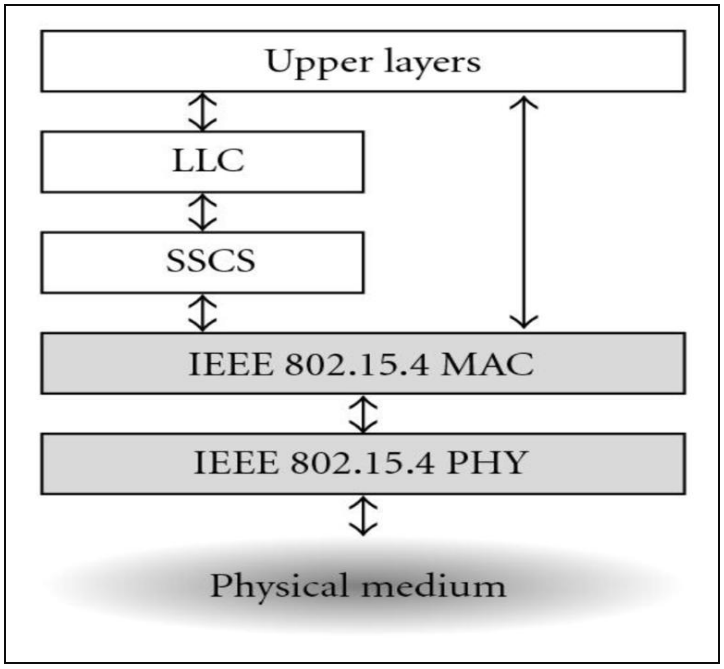
Disadvantages of IEEE 802.15.4:
IEEE 802.15.4's drawbacks include:

- IEEE 802.15.4 causes interference and multipath fading.
- doesn't employ a frequency-hopping approach.
- unbounded latency
- interference susceptibility

Applications of IEEE 802.15.4:
IEEE 802.15.4 Applications:

- Wireless sensor networks in the industry
- Building and home automation
- Remote controllers and interacting toys
- Automotive networks

The architecture of LR-WPAN Device:



UNIT 3: IP LAYER

1. What is the need for optimization in IP?

- Internet of Things will largely be built on the Internet Protocol suite
- In coping with the integration of non-IP devices, may need to deal with the limits at the device and network levels that IoT often imposes.
- Therefore, optimizations are needed at various layers of the IP stack to handle the restrictions that are present in IoT networks.
- The following concepts take a detailed look at why optimization is necessary for IP:
- Constrained Nodes
- Constrained Networks
- IP Versions

Constrained Nodes :

- IoT having different classes of devices coexist.
- Depending on its functions in a network, a "thing" architecture may or may not offer similar characteristics compared to a generic PC or server in an IT environment.
- Another limit is that this network protocol stack on an IoT node may be required to communicate through an unreliable path.
- IoT constrained nodes can be classified as follows:
  - **Devices that are very constrained in resources, may communicate infrequently to transmit a few bytes, and may have limited security and management capabilities:**
    This drives the need for the IP adaptation model, where nodes communicate through gateways and proxies.
  - **Devices with enough power and capacities to implement a strippeddown IP stack or non-IP stack:**
    In this case, you may implement either an optimized IP stack and directly communicate with application servers (adoption model) or go for an IP or non-IP stack and communicate

through gateways and proxies (adaptation model).

- **Devices that are similar to generic PCs in terms of computing and power resources but have constrained networking capacities, such as bandwidth:**

These nodes usually implement a full IP stack (adoption model), but network design and application behaviors must cope with the bandwidth constraints.

- In constrained nodes, the costs of computing power, memory, storage resources, and power consumption are generally decreasing.
- At the same time, networking technologies continue to improve and offer more bandwidth and reliability.

Constrained Networks :

- Low-speed connections (like low-speed modems) demonstrated that IP could run over low-bandwidth networks.
- High-speed connections are not usable by some IoT devices in the last mile.
- The reasons include the implementation of technologies with low bandwidth, limited distance and bandwidth due to regulated transmit power, and lack of or limited network services.
- A constrained network can have high latency and a high potential for packet loss.
- Constrained networks are often referred to as low-power and lossy networks (LLNs).
- Constrained networks operate between a few kbps and a few hundred kbps and may utilize a star, mesh, or combined network topologies, ensuring proper operations.
- In constrained network, it is not unusual for the packet delivery rate (PDR) to oscillate between low and high percentages.
- Large bursts of unpredictable errors and even loss of connectivity at times may occur, where packet delivery variation may fluctuate greatly during the course of a day.

Latency and control plane reactivity:

- One of the golden rules in a constrained network is to "underreact to failure."
- Due to the low bandwidth, a constrained network that overreacts can lead to a network collapse which makes the existing problem worse.

- Control plane traffic must also be kept at a minimum; otherwise, it consumes the bandwidth that is needed by the data traffic.
- **The power consumption in battery-powered nodes:**
  - Any failure or verbose control plane protocol may reduce the lifetime of the batteries.
- This led to work on optimizing protocols for IoT

IP Versions :

- IETF has been working on transitioning the Internet from IP version 4 to IP version 6.
- The main driving force has been the lack of address space in IPv4 as the Internet has grown.
- IPv6 has a much larger range of addresses that should not be exhausted for the foreseeable future.
- Today, both versions of IP run over the Internet, but most traffic is still IPv4 based.
- Internet of Things has the Internet itself and support both IPv4 and IPv6 versions concurrently.
- The following are some of the main factors applicable to IPv4 and IPv6 support in an IoT solution:
  - Application Protocol
  - Cellular Provider and Technology
  - Serial Communications
  - IPv6 Adaptation Layer

Application Protocol:

- IoT devices implementing Ethernet or Wi-Fi interfaces can communicate over both IPv4 and IPv6, but the application protocol may dictate the choice of the IP version.
- For example, SCADA protocols such as DNP3/IP (IEEE 1815), Modbus TCP, or the IEC 60870-5-104 standards are specified only for IPv4.

- So, there are no known production implementations by vendors of these protocols over IPv6 today.
- For IoT devices with application protocols defined by the IETF, such as HTTP/HTTPS, CoAP, MQTT, and XMPP, both IP versions are supported.
- The selection of the IP version is only dependent on the implementation.
- **Cellular Provider and Technology:**
  - IoT devices with cellular modems are dependent on the generation of the cellular technology as well as the data services offered by the provider.
  - For the first three generations of data services GPRS, Edge, and 3G IPv4 is the base protocol version.
  - Consequently, if IPv6 is used with these generations, it must be tunneled over IPv4.
  - On 4G/LTE networks, data services can use IPv4 or IPv6 as a base protocol, depending on the provider.
- **Serial Communications:**
  - Data is transferred using either proprietary or standards-based protocols, such as DNP3, Modbus, or IEC 60870-5-101.
  - In the past, communicating this serial data over any sort of distance could be handled by an analog modem connection.
  - However, as service provider support for analog line services has declined, the solution for communicating with these legacy devices has been to use local connections.
  - To make this work, you connect the serial port of the legacy device to a nearby serial port on a piece of communications equipment, typically a router.
  - This local router then forwards the serial traffic over IP to the central server for processing.
  - Encapsulation of serial protocols over IP leverages mechanisms such as raw socket TCP or UDP.
  - While raw socket sessions can run over both IPv4 and IPv6, current implementations are mostly available for IPv4 only
- **IPv6 Adaptation Layer:**
  - IPv6-only adaptation layers for some physical and data link layers for recently standardized IoT protocols support only IPv6.
  - While the most common physical and data link layers (Ethernet, Wi-Fi, and so on) stipulate adaptation layers for both versions, newer technologies, such as IEEE 802.15.4 (Wireless Personal Area Network), IEEE 1901.2, and ITU G.9903 (Narrowband Power Line Communications) only have an IPv6 adaptation layer specified.
  - This means that any device implementing a technology that requires an IPv6 adaptation layer must communicate over an IPv6-only subnet work.
  - This is reinforced by the IETF routing protocol for LLNs, RPL, which is IPv6 only.

2. What is the need for optimizing IP for IoT?

- While the Internet Protocol is key for a successful Internet of Things, constrained nodes and constrained networks mandate optimization at various layers and on multiple protocols of the IP architecture

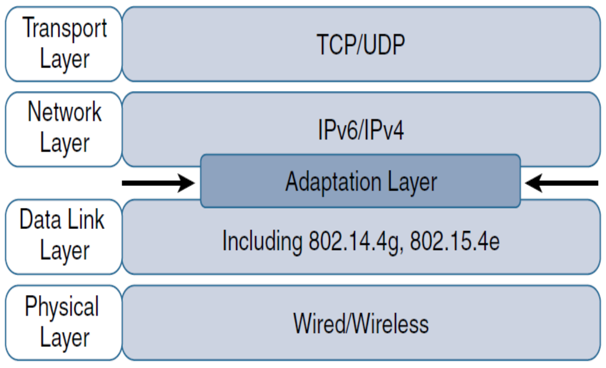


Figure 5-1 *Optimizing IP for IoT Using an Adaptation Layer*

The following optimizations technique of IP already available:

- From 6LoWPAN to 6Lo
  - Header Compression
  - Fragmentation
  - Mesh Addressing

    - Mesh-Under Versus Mesh-Over Routing
    - 6Lo Working Group
- 6TiSCH
- RPL
    - Objective Function (OF)
    - Rank
    - RPL Headers
    - Metrics
    - Authentication and Encryption on Constrained Nodes
    - ACE
    - DICE

From 6LoWPAN to 6Lo :

- In the IP architecture, the transport of IP packets over any given Layer 1 (PHY) and Layer 2 (MAC) protocol must be defined.
- The initial focus of the 6LoWPAN working group was to optimize the transmission of IPv6 packets over constrained networks such as IEEE 802.15.4.
- IoT-related protocols follow a similar process.
- The main difference is that an adaptation layer designed for IoT may include some optimizations to deal with constrained nodes and networks.

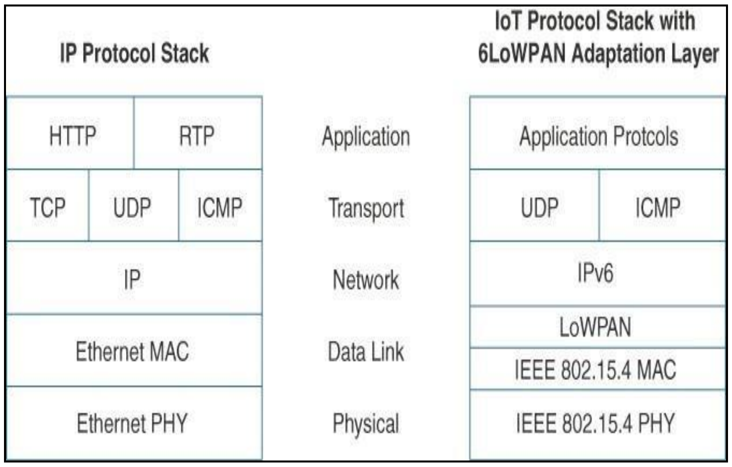


Fig.*Comparison of an IoT Protocol Stack Utilizing 6LoWPAN and an IP Protocol Stack*

- The 6LoWPAN working group published several RFCs (Request for Comments by IETF), but RFC defines frame headers for the capabilities of
    - Header compression,
    - Fragmentation,
    - Mesh addressing.
- These headers can be stacked in the adaptation layer to keep these concepts separate while enforcing a structured method for expressing each capability.
- Depending on the implementation, all, none, or any combination of these capabilities and their corresponding headers can be enabled.

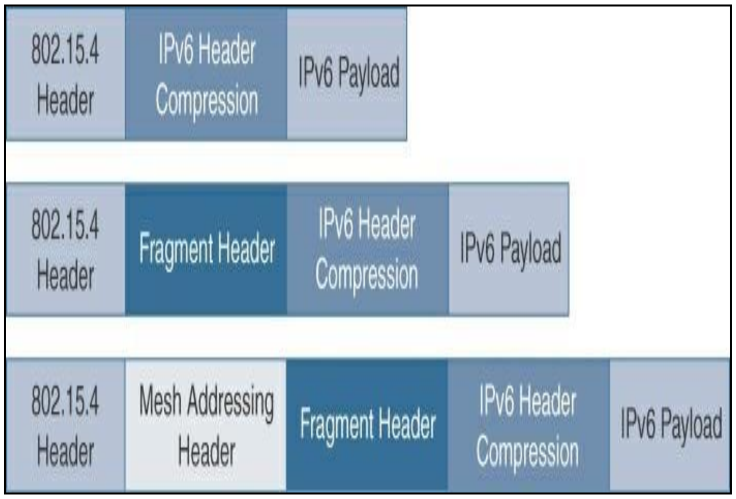


Figure 3.3 *6LoWPAN Header Stacks*

Header Compression:

- Shrinks the size of IPv6's 40-byte headers and User Datagram Protocol's (UDP's) 8-byte headers down as low as 6 bytes combined in some cases.
- Header compression for 6LoWPAN is only defined for an IPv6 header and not for IPv4.
- However, a number of factors affect the amount of compression, such as implementation of RFC, whether UDP is included, and various IPv6 addressing scenarios.

- 6LoWPAN works by taking advantage of shared information known by all nodes from their participation in the local network.
- In addition, it omits some standard header fields by assuming commonly used values.

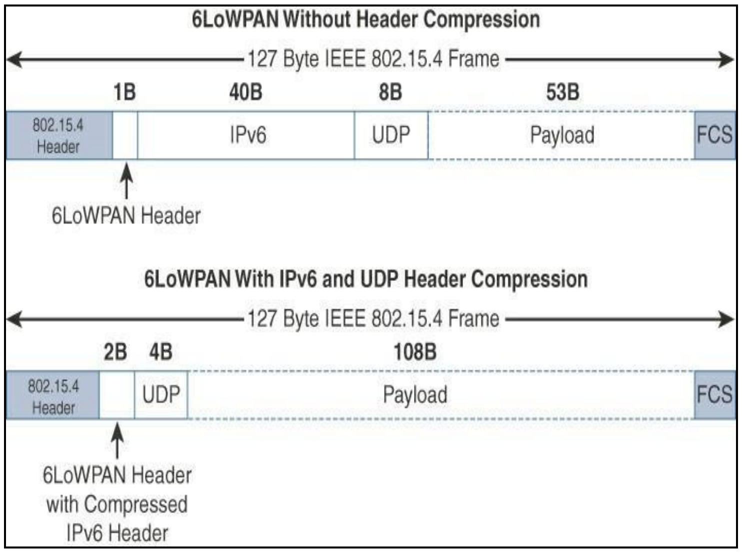

6LoWPAN Without Header Compression
127 Byte IEEE 802.15.4 Frame
1B
40B
8B
53B
802.15.4 Header
IPv6
UDP
Payload
FCS
6LoWPAN Header
6LoWPAN With IPv6 and UDP Header Compression
127 Byte IEEE 802.15.4 Frame
2B
4B
108B
802.15.4 Header
UDP
Payload
FCS
6LoWPAN Header with Compressed IPv6 Header

Figure 3.4 *6LoWPAN Header Compression*

- At the top of, a 6LoWPAN frame without any header compression enabled:
  - The full 40-byte IPv6 header and 8-byte UDP header are visible.
  - The 6LoWPAN header is only a single byte in this case.
  - Uncompressed IPv6 and UDP headers leave only 53 bytes of data payload out of the 127-byte maximum frame size in the case of IEEE 802.15.4.
- The bottom half of shows a frame where header compression enabled:
  - The 6LoWPAN header increases to 2 bytes to accommodate the compressed IPv6 header, and UDP has been reduced in half, to 4 bytes from 8.
  - Most importantly, the header compression has allowed the payload to more than double, from 53 bytes to 108 bytes, which is obviously much more efficient.

Fragmentation:

- The maximum transmission unit (MTU) for an IPv6 network must be at least 1280 bytes.
- The term MTU defines the size of the largest protocol data unit that can be passed.
- For IEEE 802.15.4, 127 bytes is the MTU.
- A problem because of IPv6, with a much larger MTU, is carried inside the 802.15.4 frame with a much smaller one.
- To remedy this situation, large IPv6 packets must be fragmented across multiple 802.15.4 frames at Layer 2.

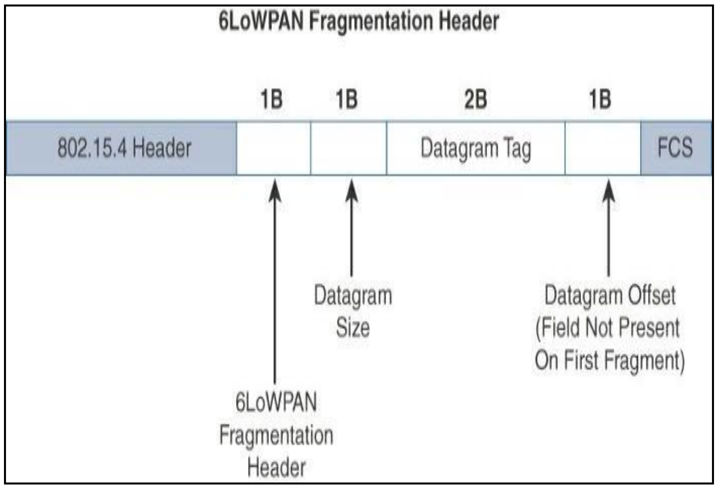
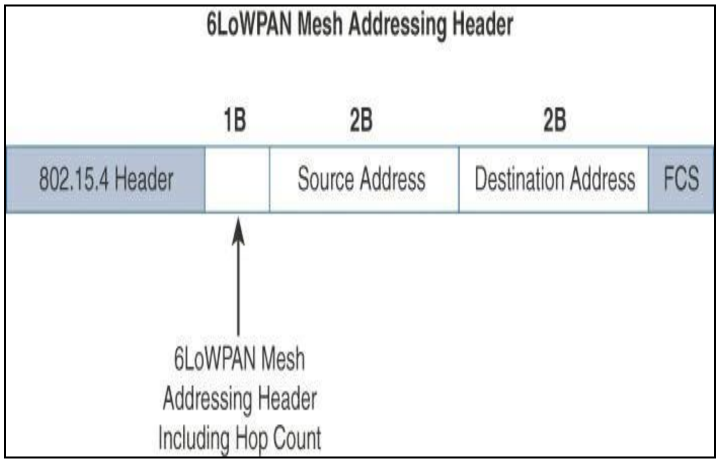

6LoWPAN Fragmentation Header
1B
1B
2B
1B
802.15.4 Header
Datagram Tag
FCS
Datagram Size
Datagram Offset (Field Not Present On First Fragment)
6LoWPAN Fragmentation Header

Figure 3.5 *6LoWPAN Fragmentation Header*

- The fragment header utilized by 6LoWPAN is composed of three primary fields:
  - **Datagram Size:**
    The 1-byte field specifies the total size of the unfragmented payload
  - **Datagram Tag:**
    identifies the set of fragments for a payload.
  - **Datagram Offset:**
    field delineates how far into a payload a particular fragment occurs.
- The 6LoWPAN fragmentation header field itself uses a unique bit value to identify that the subsequent fields behind it are fragment fields as opposed to another capability, such as header compression.
- In the first fragment, the Datagram Offset field is not present because it would simply be set to 0.
- This results in the first fragmentation header for an IPv6 payload being only 4 bytes long.

Mesh Addressing:

- The purpose of the 6LoWPAN mesh addressing function is to forward packets over multiple hops.
- Three fields are defined for this header:


6LoWPAN Mesh Addressing Header
1B
2B
2B
802.15.4 Header
Source Address
Destination Address
FCS
6LoWPAN Mesh Addressing Header Including Hop Count

- **Hop Limit:**

  The hop limit for mesh addressing also provides an upper limit on how many times the frame can be forwarded.

  Each hop decrements this value by 1 as it is forwarded. Once the value hits 0, it is dropped and no longer forwarded.
- **Source Address, and Destination Address:**

  The Source Address and Destination Address fields for mesh addressing are IEEE 802.15.4 addresses indicating the endpoints of an IP hop.

Mesh-Under Versus Mesh-Over Routing:

- IEEE 802.15.4, IEEE 802.15.4g, and IEEE 1901.2a that support mesh topologies and operate at the physical and data link layers, two main options exist for establishing reachability and forwarding packets.
- **"Mesh-under":** The routing of packets is handled at the 6LoWPAN adaptation layer.
- **"Mesh-over" or "route-over":** utilizes IP routing for getting packets to their destination.
- Mesh-under routing,
- the routing of IP packets leverages the 6LoWPAN mesh addressing header to route and forward packets at the link layer.
- The term mesh-under is used because multiple link layer hops can be used to complete a single IP hop.
- Nodes have a Layer 2 forwarding table that they consult to route the packets to their final destination within the mesh.
- An edge gateway terminates the mesh-under domain.
- The edge gateway must also implement a mechanism to translate between the configured Layer 2 protocol and any IP routing mechanism implemented on other Layer 3 IP interfaces.
- Mesh-over or route-over scenarios,
  - IP Layer 3 routing is utilized for computing reachability and then getting packets forwarded to their destination, either inside or outside the mesh domain.
  - Each full-functioning node acts as an IP router, so each link layer hop is an IP hop.
  - When a LoWPAN has been implemented using different link layer technologies, a mesh-over routing setup is useful.
  - While traditional IP routing protocols can be used, a specialized routing protocol for smart objects, such as RPL

6Lo Working Group:

- The 6Lo working group seeks to expand on this completed work with focus on IPv6 connectivity over constrained node networks.
- While the 6LoWPAN working group initially focused its optimizations on IEEE 802.15.4 LLNs,
- 6Lo working group is focused on the following:

IPv6-over-foo adaptation layer specifications using 6LoWPAN technologies (RFC4944, RFC6282, RFC6775) for link layer technologies:

- For example, this includes:
- IPv6 over Bluetooth Low Energy
- Transmission of IPv6 packets over near-field communication
- IPv6 over 802.11ah
- Transmission of IPv6 packets over DECT Ultra Low Energy
- Transmission of IPv6 packets on WIA-PA (Wireless Networks for Industrial Automation–Process Automation)
- Transmission of IPv6 over Master Slave/Token Passing (MS/TP)

Information and data models such as MIB modules:

- One example is RFC 7388, "Definition of Managed Objects for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)."

Optimizations that are applicable to more than one adaptation layer specification:

- For example, this includes RFC 7400, "6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)."
- Informational and maintenance publications needed for the IETF specifications in this area.

6TiSCH

- IEEE 802.15.4e, Time-Slotted Channel Hopping (TSCH), is an add-on to the Media Access Control (MAC) portion of the IEEE 802.15.4 standard,
- Devices implementing IEEE 802.15.4e TSCH communicate by following a Time Division Multiple Access (TDMA) schedule.
- An allocation of a unit of bandwidth or time slot is scheduled between neighbor nodes.
- This allows the programming of predictable transmissions and enables deterministic, industrial-type applications.
- Not like other IEEE 802.15.4
- To standardize IPv6 over the TSCH mode of IEEE 802.15.4e (known as 6TiSCH)
- The IEEE 802.15.4e standard defines a time slot structure, but it does not mandate a scheduling algorithm for how the time slots are utilized.
- This is left to higher-level protocols like 6TiSCH.
- Scheduling is critical because it can affect throughput, latency, and power consumption.

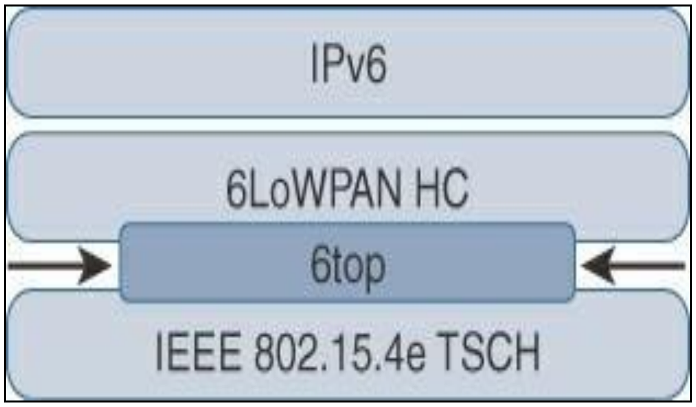


Fig. 5-7 *Location of 6TiSCH's 6top Sublayer*

- Schedules in 6TiSCH are broken down into cells.
- A cell is simply a single element in the TSCH schedule that can be allocated for unidirectional or bidirectional communications between specific nodes. •
- Nodes only transmit when the schedule dictates that their cell is open for communication.
- The 6TiSCH architecture defines four schedule management mechanisms: • Static scheduling
- Neighbor-to-neighbor scheduling
- Remote monitoring and scheduling management
- Hop-by-hop scheduling

Static scheduling:

- All nodes in the constrained network share a fixed schedule.
- Cells are shared, and nodes contend for slot access in a slotted aloha manner.
- Slotted aloha is a basic protocol for sending data using time slot boundaries when communicating over a shared medium.
- Static scheduling is a simple scheduling mechanism that can be used upon initial implementation or as a fall back in the case of network malfunction.
- The drawback with static scheduling is that nodes may expect a packet at any cell in the schedule.

- Therefore, energy is wasted idly listening across all cells.

Neighbor-to-neighbor scheduling:

- A schedule is established that correlates with the observed number of transmissions between nodes.
- Cells in this schedule can be added or deleted as traffic requirements and bandwidth needs change.

Remote monitoring and scheduling management:

- Time slots and other resource allocation are handled by a management entity that can be multiple hops away.
- The scheduling mechanism leverages 6top and even CoAP in some scenarios.
- This scheduling mechanism provides quite a bit of flexibility and control in allocating cells for communication between nodes.

Hop-by-hop scheduling:

- A node reserves a path to a destination node multiple hops away by requesting the allocation of cells in a schedule at each intermediate node hop in the path.
- The protocol that is used by a node to trigger this scheduling mechanism is not defined at this point.
- In addition to schedule management functions, the 6TiSCH architecture also defines three different forwarding models.
- Forwarding is the operation performed on each packet by a node that allows it to be delivered to a next hop or an upperlayer protocol.
- The forwarding decision is based on a pre existing state that was learned from a routing computation.
- There are three 6TiSCH forwarding models:

1. Track Forwarding (TF)
2. Fragment forwarding (FF)
3. IPv6 Forwarding (6F)

1. **Track Forwarding (TF):**
   - This is the simplest and fastest forwarding model.
   - A “track” in this model is a unidirectional path between a source and a destination.
   - This track is constructed by pairing bundles of receive cells in a schedule with a bundle of receive cells set to transmit.
   - So, a frame received within a particular cell or cell bundle is switched to another cell or cell bundle.
   - This forwarding occurs regardless of the network layer protocol.
2. **IPv6 Forwarding (6F):**
   - 2This model forwards traffic based on its IPv6 routing table.
   - Flows of packets should be prioritized by traditional QoS (quality of service) and RED (random early detection) operations.
   - QoS is a classification scheme for flows based on their priority, and RED is a common congestion avoidance mechanism.
3. **Fragment forwarding (FF):**
   - This model takes advantage of 6LoWPAN fragmentation to build a Layer 2 forwarding table.
   - Fragmentation within the 6LoWPAN protocol.
   - IPv6 packets can get fragmented at the 6LoWPAN sublayer to handle the differences between IEEE 802.15.4 payload size and IPv6 MTU.
   - Additional headers for RPL source route information can further contribute to the need for fragmentation.
   - However, with FF, a mechanism is defined where the first fragment is routed based on the IPv6 header present.
   - This increases latency and can be power- and CPU-intensive for a constrained node.

RPL

- IETF chartered the RoLL (Routing over Low-Power and Lossy Networks) working group to evaluate all Layer 3 IP routing protocols and determine the needs and requirements for developing a routing solution for IP smart objects.

- The new routing protocol should be developed for use by IP smart objects is IPv6 Routing Protocol for Low Power and Lossy Networks (RPL).
- In an RPL network,
  - each node acts as a router and becomes part of a mesh network.
  - Routing is performed at the IP layer.
  - Each node examines every received IPv6 packet and determines the nexthop destination based on the information contained in the IPv6 header.
- The constraints of computing and memory that are common characteristics of constrained nodes, the protocol defines two modes:

Storing mode:

- All nodes contain the full routing table of the RPL domain. Every node knows how to directly reach every other node.

Non-storing mode:

- Only the border router(s) of the RPL domain contain(s) the full routing table.
- All other nodes in the domain only maintain their list of parents and use this as a list of default routes toward the border router.
- This abbreviated routing table saves memory space and CPU.
- When communicating in non-storing mode, a node always forwards its packets to the border router, which knows how to ultimately reach the final destination.

- RPL is based on the concept of a directed acyclic graph (DAG). A DAG is a directed graph where no cycles exist.
- This means that from any vertex or point in the graph, you cannot follow an edge or a line back to this same point.
- All of the edges are arranged in paths oriented toward and terminating at one or more root nodes.

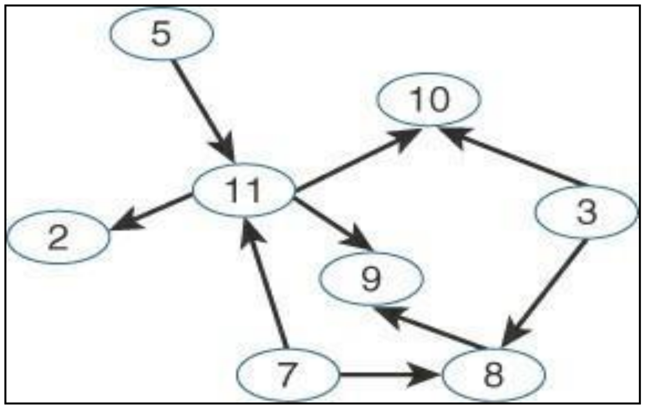


Fig. *Example of a Directed Acyclic Graph (DAG)*

- A basic RPL process involves building a destination-oriented directed acyclic graph (DODAG).
- A DODAG is a DAG rooted to one destination.
- In RPL, this destination occurs at a border router known as the DODAG root.
- Observe that that a DAG has multiple roots, whereas the DODAG has just one.

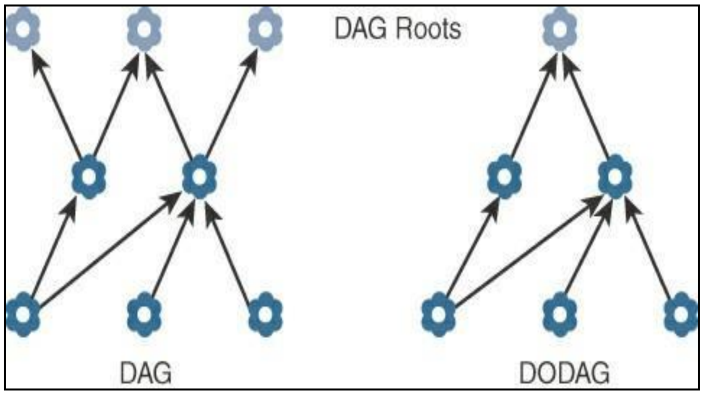


Fig.*DAG and DODAG Comparison*

- In a DODAG, each node maintains up to three parents that provide a path to the root.
- one of these parents is the preferred parent, which means it is the preferred next hop for upward routes toward the root.
- The routing graph created by the set of DODAG parents across all nodes defines the full set of upward routes.
- RPL protocol implementation should ensure that routes are loop free by disallowing nodes from selected DODAG parents that are positioned further away from the border router.

- Upward routes in RPL are discovered and configured using DAG Information Object (DIO) messages.
- Nodes listen to DIOs to handle changes in the topology that can affect routing.
- Nodes establish downward routes by advertising their parent set toward the DODAG root using a Destination Advertisement Object (DAO) message.
- DAO messages allow nodes to inform their parents of their presence and reachability to descendants.
- In non-storing mode of RPL,
  nodes sending DAO messages report their parent sets directly to the DODAG root (border router), and only the root stores the routing information.
  - In storing mode,
    each node keeps track of the routing information that is advertised in the DAO messages.
  - The nodes can make their own routing decisions; in non-storing mode, on the other hand, all packets must go up to the root to get a route for moving downstream.
  - RPL messages, such as DIO and DAO, run on top of IPv6.

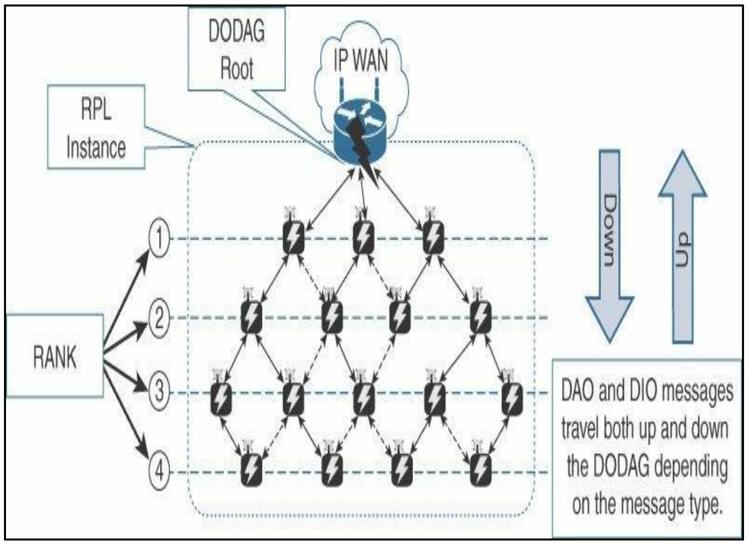

DODAG Root
IP WAN
RPL Instance
RANK
1
2
3
4
Down
Up
DAO and DIO messages travel both up and down the DODAG depending on the message type.

Fig. *RPL Overview*

Objective Function (OF) :

- An objective function (OF) defines how metrics are used to select routes and establish a node's rank.
- For example, nodes implementing an OF with Minimum Expected Number of Transmissions (METX) advertise the METX among their parents in DIO messages.
- Whenever a node establishes its rank, it simply sets the rank to the current minimum METX among its parents.

Rank

- The rank is a rough approximation of how "close" a node is to the root and helps avoid routing loops and the count-to-infinity problem.
- Nodes can only increase their rank when receiving a DIO message with a larger version number.
- However, nodes may decrease their rank whenever they have established lower-cost routes.
- While the rank and routing metrics are closely related, the rank differs from routing metrics in that it is used as a constraint to prevent routing loops.

RPL Headers:

- An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL).
- A new IPv6 option, known as the RPL option
- The RPL option is carried in the IPv6 Hop-by-Hop header.
- The purpose of this header is to leverage data plane packets for loop detection in a RPL instance.
- Source Routing Header (SRH) for use between RPL routers.
- A border router or DODAG root inserts the SRH when specifying a source route to deliver datagrams to nodes downstream in the mesh network.

Metrics

- Some of the RPL routing metrics and constraints defined in RFC 6551 include the following
- Expected Transmission Count (ETX):

Assigns a discrete value to the number of transmissions a node expects to make to deliver a packet.

- Hop Count:

Tracks the number of nodes traversed in a path. Typically, a path with a lower hop count is chosen over a path with a higher hop count.

- Latency:

Varies depending on power conservation. Paths with a lower latency are preferred.

- Link Quality Level:

Measures the reliability of a link by taking into account packet error rates caused by factors such as signal attenuation and interference.

- Link Color:

Allows manual influence of routing by administratively setting
values to make a link more or less desirable. These values can be either statically or dynamically adjusted for specific traffic types.

- Node State and Attribute:

Identifies nodes that function as traffic aggregators and nodes that are being impacted by high workloads.
High workloads could be indicative of nodes that have incurred high CPU or low memory states. Naturally, nodes that are aggregators are preferred over nodes experiencing high workloads.

- Node Energy:

Avoids nodes with low power, so a battery-powered node that is running out of energy can be avoided and the life of that node and the network can be prolonged.

- Throughput:

Provides the amount of throughput for a node link.
Often, nodes conserving power use lower throughput.
This metric allows the prioritization of paths with higher throughput.
Authentication and Encryption on Constrained Nodes.

- The IETF working groups that are focused on IoT security:
- ACE and DICE.

ACE:

- The Authentication and Authorization for Constrained Environments (ACE) working group is tasked with evaluating the applicability of existing authentication and authorization protocols and documenting their suitability for certain constrained-environment use cases.
- ACE working group will focus its work on CoAP with the Datagram Transport Layer Security (DTLS) protocol.
- The ACE working group expects to produce a standardized solution for authentication and authorization that enables authorized access (Get, Put, Post, Delete) to resources identified by a URI and hosted on a resource server in constrained environments.
- An unconstrained authorization server performs mediation of the access.

DICE:

- New generations of constrained nodes implementing an IP stack over constrained access networks are expected to run an optimized IP protocol stack.
- The DTLS in Constrained Environments (DICE) working group focuses on implementing the DTLS transport layer security protocol in these environments.
- The first task of the DICE working group is to define an optimized DTLS profile for constrained nodes.
- In addition, the DICE working group is considering the applicability of the DTLS record layer to secure multicast messages and investigating how the DTLS handshake in constrained environments can get optimized.

3. Explain SCADA protocol in detail ?

SCADA

- Supervisory control and data acquisition (SCADA).
- Designed decades ago, SCADA is an automation control system that was initially implemented without IP over serial links (such

as RS-232 and RS-485), before being adapted to Ethernet and IPv4.

A Little Background on SCADA

- SCADA networking protocols, running directly over serial physical and data link layers.
- At a high level, SCADA systems collect sensor data and telemetry from remote devices, and to control them.
- SCADA systems allow global, real-time, data-driven decisions to be made about how to improve business processes.
- SCADA commonly uses certain protocols for communications between devices and applications.
- E.g.Modbus industrial protocols used to monitor and program remote devices via a master/slave relationship.
- Modbus used in building management, transportation, and energy applications.
- The DNP3 (Distributed Network Protocol) and International Electrotechnical Commission (IEC) protocols are found mainly in the utilities industry, along with DLMS/COSEM
- ANSI C12 for advanced meter reading (AMR).
- These protocols used decade ago and are serial based. So, transporting them over current IoT and traditional networks requires that certain Adjustments.

Adapting SCADA for IP 92

- Ethernet and IP include the ability to leverage existing equipment and standards while integrating seamlessly the SCADA subnetworks to the corporate WAN infrastructures.
- Assigning TCP/UDP to the protocols, as following:
  - DNP3 (adopted by IEEE 1815-2012) specifies the use of TCP or UDP on port 20000 for transporting DNP3 messages over IP.
  - The Modbus messaging service utilizes TCP.
  - IEC 60870-5-104 is the evolution of IEC 60870-5-101 serial for running over Ethernet and IPv4 using port 2404.
  - DLMS User Association specified a communication profile based on TCP/IP in the DLMS/COSEM.
- Serial protocols have adapted and evolved to utilize IP and TCP/UDP as both networking and transport mechanisms.
- DNP3 (Distributed Network Protocol) is based on a master/slave relationship.
- The term master is refers to a powerful computer located in the control center of a utility, and a slave is a remote device with computing resources found in a location such as a substation.
- DNP3 refers to slaves as outstations.
- Outstations monitor and collect data from devices that indicate their state, such as whether a circuit breaker is on or off, and take measurements, including voltage, current, temperature, and so on.
- This data is then transmitted to the master when it is requested, or events or alarms and control commands can be sent in an asynchronous manner.

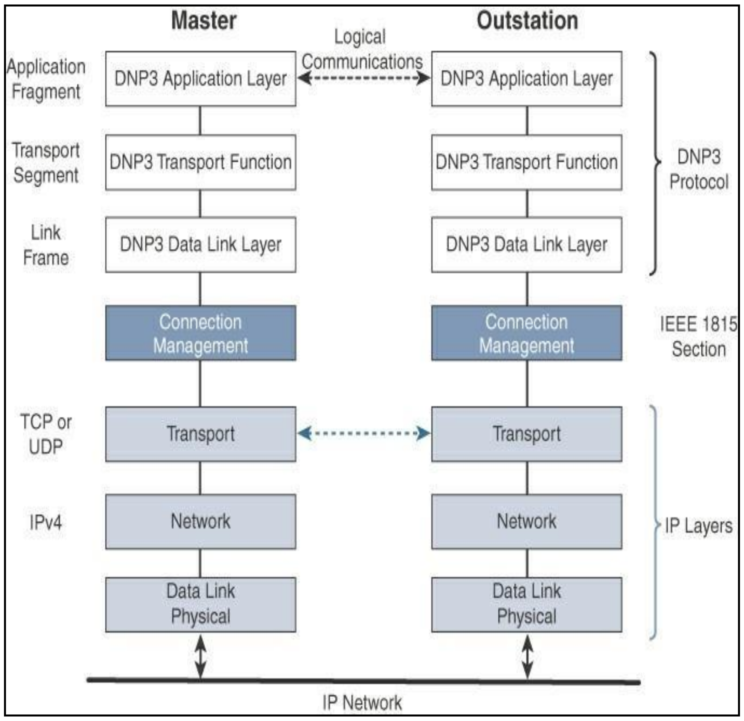
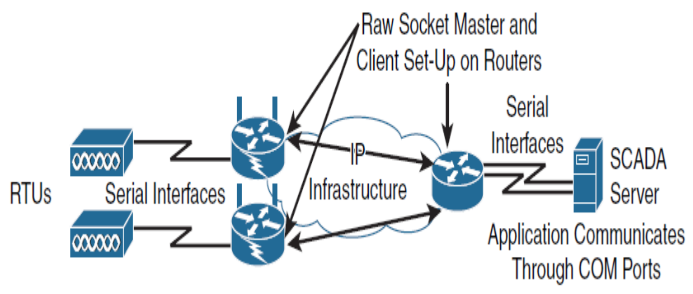


Figure 3.12 *Protocol Stack for Transporting Serial DNP3 SCADA over IP*

- Connection management links the DNP3 layers with the IP layers in addition to the configuration parameters and methods necessary for implementing the network connection.
- The master side initiates connections by performing a TCP active open.
- The outstation listens for a connection request by performing a TCP passive open.
- Master stations may parse multiple DNP3 data link layer frames from a single UDP datagram, while DNP3 data link layer frames cannot span multiple UDP datagrams.

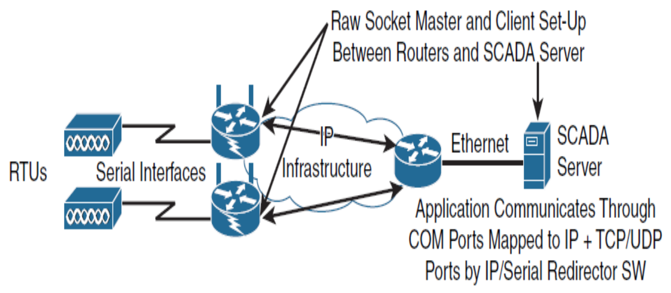
Tunneling Legacy SCADA over IP Networks

- End-to-end native IP support is preferred, in the case of DNP3.
- Otherwise, transport of the original serial protocol over IP can be achieved either by tunneling using raw sockets over TCP or UDP or by installing an intermediate device that performs protocol translation between the serial protocol version and its IP implementation.
- A raw socket connection simply denotes that the serial data is being packaged directly into a TCP or UDP transport.
- A socket is a standard application programming interface (API) composed of an IP address and a TCP or UDP port that is used to access network devices over an IP network.
- Scenarios A, B and C in Figure,
- Routers connect via serial interfaces to the remote terminal units (RTUs), which are often associated with SCADA networks.
- An RTU is a multipurpose device used to monitor and control various systems, applications, and devices managing automation.
- From the master/slave perspective, the RTUs are the slaves.
- Opposite the RTUs in is a SCADA server, or master, that varies its connection type



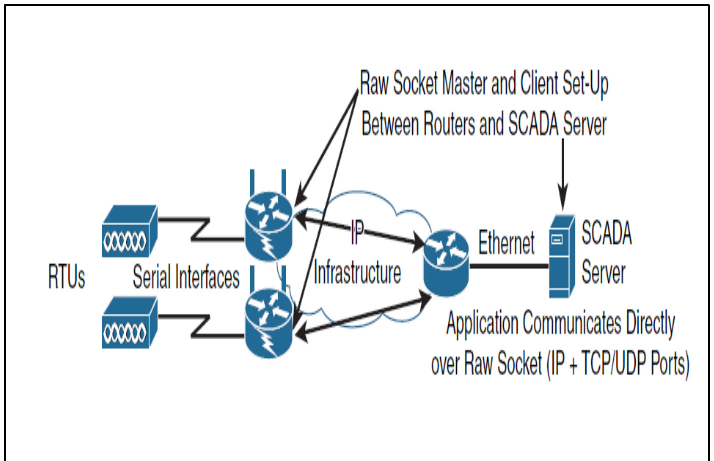
Scenario A: Raw Socket between Routers – no change on SCADA server

- **Scenarios A:**
- Both the SCADA server and the RTUs have a direct serial connection to their respective routers.
- The routers terminate the serial connections at both ends of the link and use raw socket encapsulation to transport the serial payload over the IP network.


Raw Socket Master and Client Set-Up
Between Routers and SCADA Server
IP
Infrastructure
Ethernet
SCADA
Server
RTUs
Serial Interfaces
Application Communicates Through
COM Ports Mapped to IP + TCP/UDP
Ports by IP/Serial Redirector SW

Scenario B: Raw Socket between Router and SCADA Server – no SCADA application change on server but IP/Serial Redirector software and Ethernet interface to be added

- **Scenarios B**:
- A small change on the SCADA server side. A piece of software is installed on the SCADA server that maps the serial COM ports to IP ports.
- This software is commonly referred to as an IP/serial redirector.
- The IP/serial redirector in essence terminates the serial connection of the SCADA server and converts it to a TCP/IP port using a raw socket connection.



Raw Socket Master and Client Set-Up Between Routers and SCADA Server
RTUs
Serial Interfaces
IP Infrastructure
Ethernet
SCADA Server
Application Communicates Directly over Raw Socket (IP + TCP/UDP Ports)

Scenario C: Raw Socket between Router and SCADA Server – SCADA application knows how to directly communicate over a Raw Socket and Ethernet interface

Scenarios C:

- The SCADA server supports native raw socket capability.
- Unlike in Scenarios A and B, where a router or IP/serial redirector software has to map the SCADA server's serial ports to IP ports, in Scenario C the SCADA server has full IP support for raw socket connections.

SCADA Protocol Translation

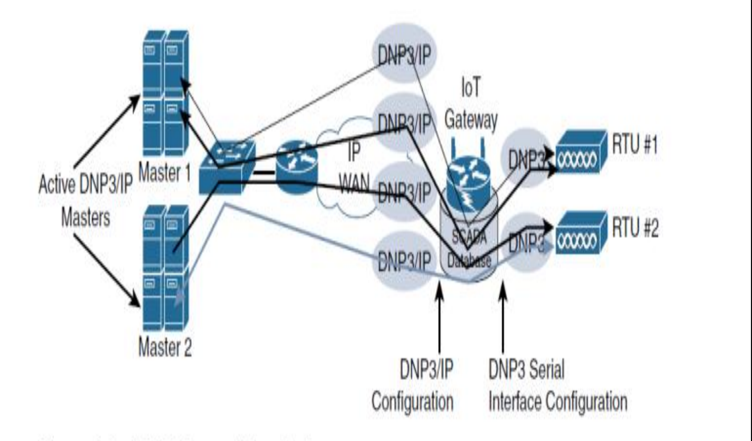


Figure 6-4 *DNP3 Protocol Translation*

- With protocol translation, the legacy serial protocol is translated to a corresponding IP version.Figure shows two serially connected DNP3 RTUs and two master applications supporting DNP3 over IP that control and pull data from the RTUs.
- **The IoT gateway in this figure performs a protocol translation function that enables communication between the RTUs and servers, despite the fact that a serial connection is present on one side and an IP connection is used on the other**.
- By running protocol translation, the IoT gateway connected to the RTUs is implementing a computing function close to the edge of the network.
- Adding computing functions close to the edge helps scale distributed intelligence in IoT networks.

SCADA Transport over LLNs with MAP-T

- Due to the constrained nature of LLNs, the implementation of industrial protocols should at a minimum be done over UDP.
- This in turn requires that both the application servers and devices support and implement UDP.
- When deployed over LLN subnetworks that are IPv6 only, a transition mechanism, such as MAP-T (Mapping of Address and Port using Translation), needs to be implemented.

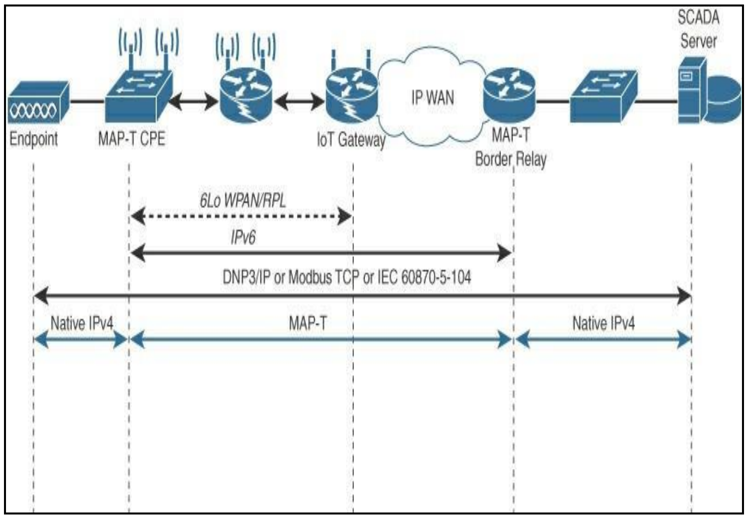


Fig. *DNP3 Protocol over 6LoWPAN Networks with MAP-T*

- In figure shows a scenario in which a legacy endpoint is connected across an LLN running 6LoWPAN to an IP-capable SCADA server.
- The legacy endpoint could be running various industrial and SCADA protocols, including DNP3/IP, Modbus/TCP, or IEC.
- In this scenario, the legacy devices and the SCADA server support only IPv4.
- MAP-T makes the appropriate mappings between IPv4 and the IPv6 protocols.
- This allows legacy IPv4 traffic to be forwarded across IPv6 networks.

- In Figure the IPv4 endpoint on the left side is connected to a Customer Premise Equipment (CPE) device.
- The MAP-T CPE device has an IPv6 connection to the RPL mesh.
- On the right side, a SCADA server with native IPv4 support connects to a MAP-T border gateway

4. Explain the difference between CoAP and MQTT protocols ?

CoAP	**MQTT**
COAP stands for Constrained Application Protocol.	MQTT stands for Message query telemetry transport.
For communication, it uses a request-response prototype	For communication, it uses the publish-subscribe prototype
It uses asynchronous and synchronous messaging.	It uses only asynchronous mode for messaging.
It uses User Datagram Protocol (UDP)	It uses the Transmission Control Protocol (TCP).
The Heather size of CAOP is 4 bytes	The Heather size of MQTT is 2 bytes
It is RESTful based	It is not RESTful based
It does not have persistence support	It is mainly used for live communication and has persistence support
It will give labels to the messages.	It does not have any such function.
It has a secured system, and its usability is in Utility area networks	It is very secure, and its usability is in IoT applications.
It has Low Latency and NAT issues.	It has Low Latency and NAT issues

5. Explain profiles and compliances for IP ?

- Profile definitions, certifications, and promotion by alliances can help implementers develop solutions that guarantee interoperability and/or interchangeability of devices.
- Some of the main industry organizations working on profile definitions and certifications for IoT constrained nodes and networks.
  - Internet Protocol for Smart Objects (IPSO) Alliance
  - Wi-SUN Alliance
  - Thread
  - IPv6 Ready Logo

Internet Protocol for Smart Objects (IPSO) Alliance

- The alliance initially focused on promoting IP as the premier solution for smart objects communications.
- Today, it is more focused on how to use IP, with the IPSO Alliance organizing interoperability tests between alliance members to validate that IP for smart objects can work together and properly implement industry standards.

Wi-SUN Alliance

- Wi-SUN's main focus is on the IEEE 802.15.4g protocol and its support for multiservice and secure IPv6 communications with applications running over the UDP transport layer.
- The utilities industry is the main area of focus for the Wi-SUN Alliance.
- The Wi-SUN field area network (FAN) profile enables smart utility networks to provide resilient, secure, and cost-effective connectivity with extremely good coverage in a range of topographic environments, from dense urban neighborhoods to rural areas.

Thread

- Thread Group has defined an IPv6-based wireless profile that provides the best way to connect more than 250 devices into a low-power, wireless mesh network.

IPv6 Ready Logo

- The IPv6 Ready Logo program has established conformance and interoperability testing programs with the intent of increasing user confidence when implementing IPv6.

- The IPv6 Core and specific IPv6 components, such as DHCP, IPsec, and customer edge router certifications, are in place.

6. Explain CoAP protocol and MQTT in detail.
CoAP

- Constrained Application Protocol (CoAP)
- is to develop a generic framework for resource-oriented applications targeting constrained nodes and networks.
- The CoAP framework defines simple and flexible ways to manipulate sensors and actuators for data or device management.
- The CoAP messaging model is primarily designed to facilitate the exchange of messages over UDP between endpoints, including the secure transport protocol Datagram Transport Layer Security (DTLS).

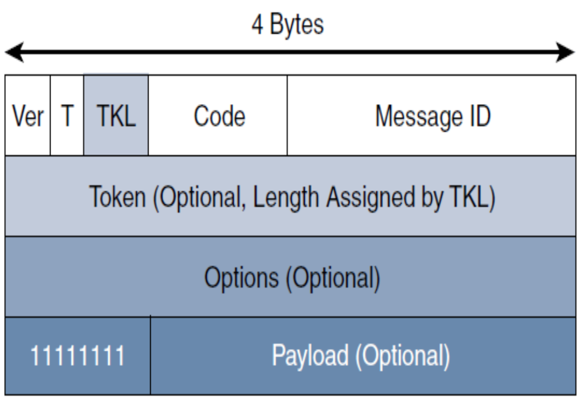


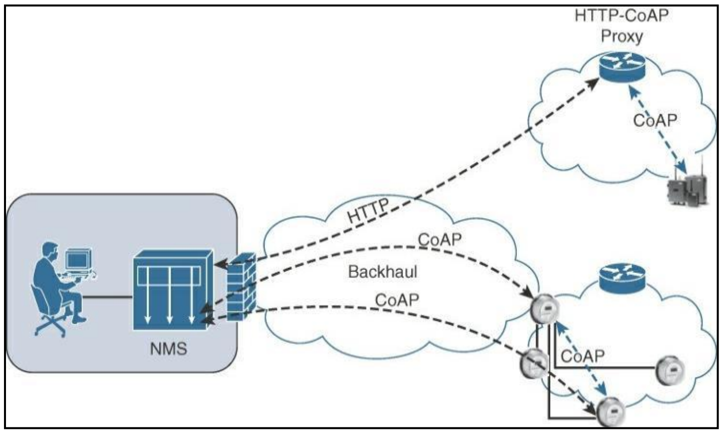
Figure 6-7 *CoAP Message Format*

- CoAP over Short Message Service (SMS) as defined in Open Mobile Alliance for Lightweight Machine-to-Machine (LWM2M) for IoT device management.
- A CoAP message is composed of
  - ~ a short fixed-length Header field (4 bytes),
  - ~ A variable-length but mandatory Token field (0–8 bytes),
  - ~ Otional field if necessary, and the payload field.

Coap message Field:

CoAP message header	Description
Ver	It is 2 bit unsigned integer. It mentions CoAP version number. Set to one.
T	It is 2 bit unsigned integer. Indicates message type viz. confirmable (0), non-confirmable (1), ACK (2) or RESET(3).
TKL	It is 4 bit unsigned integer, Indicates length of token (0 to 8 bytes)
Code	It is 8 bit unsigned integer, It is split into two parts viz. 3 bit class (MSBs) and 5 bit detail (LSBs).
Message ID	16 bit unsigned integer. Used for matching responses. Used to detect message duplication.

- CoAP can run over IPv4 or IPv6. However, it is recommended that the message fit within a single IP packet and UDP payload to avoid fragmentation.



- CoAP communications across an IoT infrastructure can take various paths.
- Connections can be between devices located on the same or different constrained networks or between devices and generic Internet or cloud servers, all operating over IP.
- As both HTTP and CoAP are IP-based protocols, the proxy function can be located practically anywhere in the network, not

necessarily at the border between constrained and non-constrained networks.

- Just like HTTP, CoAP is based on the REST architecture, but with a “thing” acting as both the client and the server.
- Through the exchange of asynchronous messages, a client requests an action via a method code on a server resource.
- A uniform resource identifier (URI) localized on the server identifies this resource.
- The server responds with a response code that may include a resource representation.
- The CoAP request/response semantics include the methods GET, POST, PUT, and DELETE.

MQTT

- Message Queuing Telemetry Transport (MQTT)
- Considering the harsh environments in the oil and gas industries, an extremely simple protocol with only a few options was designed, with considerations for constrained nodes, unreliable WAN backhaul communications, and bandwidth constraints with variable latencies.
- These were some of the rationales for the selection of a client/server and publish/subscribe framework based on the TCP/IP architecture

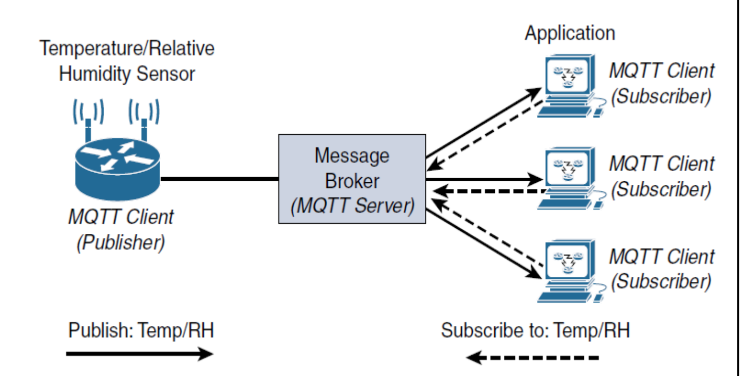


Figure 6-10 *MQTT Publish/Subscribe Framework*

- An MQTT client can
- act as a publisher to send data (or resource information) to an MQTT server acting as an MQTT message broker.
- In Figure the MQTT client on the left side is a temperature (Temp) and relative humidity (RH) sensor that publishes its Temp/RH data.

- The MQTT server (or message broker) accepts the network connection along with application messages, such as Temp/RH data, from the publishers.
- It also handles the subscription and unsubscription process and pushes the application data to MQTT clients acting as subscribers.
- The application on the right side of Figure is an MQTT client that is a subscriber to the Temp/RH data being generated by the
- publisher or sensor on the left.
- This model, where subscribers express a desire to receive information from publishers, is well known.
- The presence of a message broker in MQTT decouples the data transmission between clients acting as publishers and subscribers.
- In fact, publishers and subscribers do not even know (or need to know) about each other.
- A benefit of having this decoupling is that the MQTT message broker ensures that information can be buffered and cached in case of network failures.
- Compared to the CoAP message, MQTT contains a smaller header of 2 bytes compared to 4 bytes for CoAP.

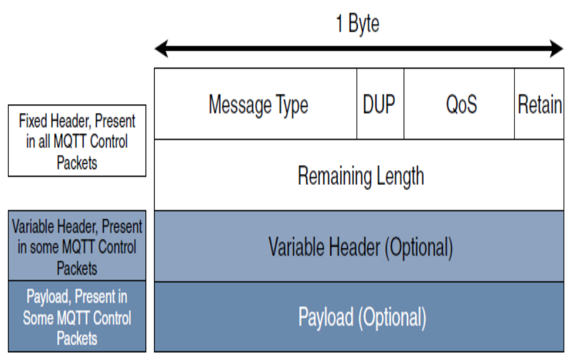
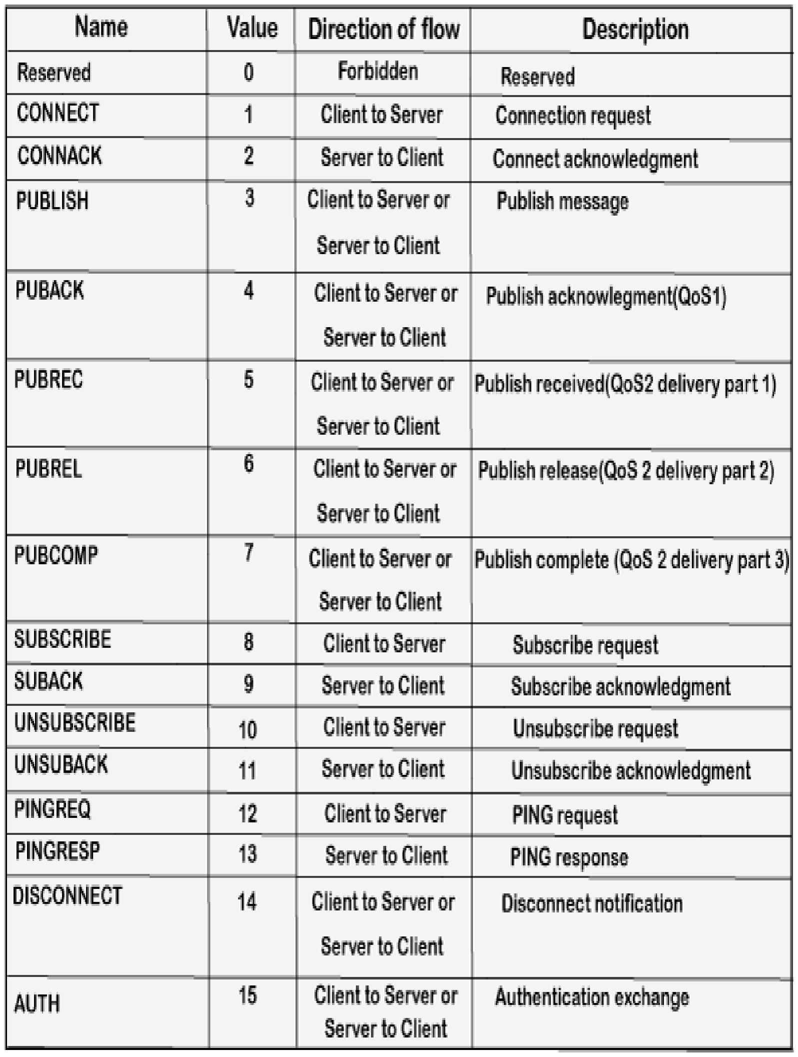

1 Byte
Message Type
DUP
QoS
Retain
Remaining Length
Variable Header (Optional)
Payload (Optional)
Fixed Header, Present in all MQTT Control Packets
Variable Header, Present in some MQTT Control Packets
Payload, Present in Some MQTT Control Packets

Figure 6-11 *MQTT Message Format*

- MQTT is a lightweight protocol because each control packet consists of a 2-byte fixed header with optional variable header fields and optional payload.

- The first MQTT field in the header is Message Type, which identifies the kind of MQTT packet within a message.
- Fourteen different types of control packets are specified in MQTT.

MQTT Control Packet Types



Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Connection request
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server or Server to Client	Publish message
PUBACK	4	Client to Server or Server to Client	Publish acknowlegment(QoS1)
PUBREC	5	Client to Server or Server to Client	Publish received(QoS2 delivery part 1)
PUBREL	6	Client to Server or Server to Client	Publish release(QoS 2 delivery part 2)
PUBCOMP	7	Client to Server or Server to Client	Publish complete (QoS 2 delivery part 3)
SUBSCRIBE	8	Client to Server	Subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server or Server to Client	Disconnect notification
AUTH	15	Client to Server or Server to Client	Authentication exchange

UNIT 4: DATA AND ANALYTICS FOR IOT

1. IoT Data Analytics Overview

- The true importance of IoT data from smart objects is realized only when the analysis of the data leads to actionable business intelligence and insights.
- Data analysis is typically broken down by the types of results that are produced

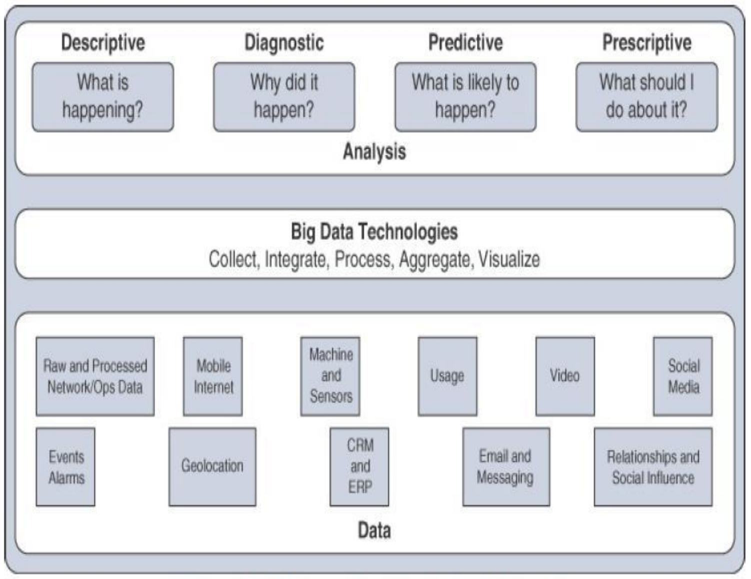


Figure 7-3 *Types of Data Analysis Results*

Four types of data analysis results :

- **Descriptive:**
  - Descriptive data analysis tells you what is happening, either now or in the past.
  - For example, a thermometer in a truck engine reports temperature values every second.
  - From a descriptive analysis perspective, you can pull this data at any moment to gain insight into the current operating condition of the truck engine.
  - If the temperature value is too high, then there may be a cooling problem or the engine may be experiencing too much load.
- **Diagnostic**:
  - When you are interested in the "why," diagnostic data analysis can provide the answer.

- Continuing with the example of the temperature sensor in the truck engine, you might wonder why the truck engine failed.
- Diagnostic analysis might show that the temperature of the engine was too high, and the engine overheated.
- Applying diagnostic analysis across the data generated by a wide range of smart objects can provide a clear picture of why a problem or an event occurred.

- **Predictive:**
  - Predictive analysis aims to foretell problems or issues before they occur.
  - For example, with historical values of temperatures for the truck engine, predictive analysis could provide an estimate on the remaining life of certain components in the engine.
  - These components could then be proactively replaced before failure occurs.
  - Or perhaps if temperature values of the truck engine start to rise slowly over time, this could indicate the need for an oil change or some other sort of engine cooling maintenance.

- **Prescriptive**:
  - Prescriptive analysis goes a step beyond predictive and recommends solutions for upcoming problems.
  - A prescriptive analysis of the temperature data from a truck engine might calculate various alternatives to cost-effectively maintain our truck
  - These calculations could range from the cost necessary for more frequent oil changes and cooling maintenance to installing new cooling equipment on the engine or upgrading to a lease on a model with a more powerful engine.
  - Prescriptive analysis looks at a variety of factors and makes the appropriate recommendation

- Both predictive and prescriptive analyses are more resource intensive and increase complexity, but the value they provide is much greater than the value from descriptive and diagnostic analysis
- You can see that descriptive analysis is the least complex and at the same time offers the least value.
- On the other end, prescriptive analysis provides the most value but is the most complex to implement.

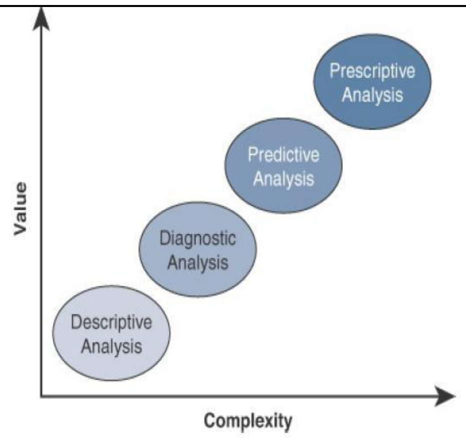


Figure 7-4 *Application of Value and Complexity Factors to the Types of Data Analysis*

IoT Data Analytics Challenges:

- **Scaling problems:**
  - Due to the large number of smart objects in most IoT networks that continually send data, relational databases can grow incredibly large very quickly.
  - This can result in performance issues that can be costly to resolve, often requiring more hardware and architecture changes.

- **Volatility of data:**
  - With relational databases, it is critical that the schema be designed correctly from the beginning.
  - Changing it later can slow or stop the database from operating.
  - Due to the lack of flexibility, revisions to the schema must be kept at a minimum.

- IoT data, however, is volatile in the sense that the data model is likely to change and evolve over time. A dynamic schema is often required so that data model changes can be made daily or even hourly.

2. Big Data Analytics Tools and Technology ?

- Big data analytics can consist of many different software pieces that together collect, store, manipulate, and analyze all different data types. Generally, the industry looks to the "three Vs" to categorize big data:

Velocity:

- *Velocity* refers to how quickly data is being collected and analyzed. Hadoop Distributed File System is designed to ingest and process data very quickly.
- Smart objects can generate machine and sensor data at a very fast rate and require database or file systems capable of equally fast ingest functions.

- **Variety:**
  - *Variety* refers to different types of data. Often you see data categorized as structured, semi-structured, or unstructured.
  - Different database technologies may only be capable of accepting one of these types.
  - Hadoop is able to collect and store all three types.
  - This can be beneficial when combining machine data from IoT devices that is very structured in nature with data from other sources, such as social media or multimedia that is unstructured.

- **Volume:**
- *Volume* refers to the scale of the data.
- Typically, this is measured from gigabytes on the very low end to petabytes or even exabytes of data on the other extreme.
- Generally, big data implementations scale beyond what is available on locally attached storage disks
- on a single node.
- It is common to see clusters of servers that consist of dozens, hundreds, or even thousands of nodes for some large deployments.
- The characteristics of big data can be defined by the sources and types of data.
  - First is machine data, which is generated by IoT devices and is typically unstructured data.
  - Second is transactional data, which is from sources that produce data from transactions on these systems, and, have high volume and structured.
  - Third is social data sources, which are typically high volume and structured.
  - Fourth is enterprise data, which is data that is lower in volume and very structured
  - Hence big data consists of data from all these separate sources.

3. Explain about MPP Shared-Nothing Architecture?

- Enterprises have used relational databases for storing structured, row and column style data types for decades.
- Relational databases are often grouped into a broad data storage category called data warehouses.
- Though they are the centerpiece of most data architectures, they are often used for longer-term archiving and data queries that can often take minutes or hours
- Massively parallel processing (MPP) databases were built on the concept of the relational data warehouses but are designed to be much faster, to be efficient, and to support reduced query times .
- To accomplish this, MPP databases take advantage of multiple nodes (computers) designed in a scale-out architecture such that both data and processing are distributed across multiple systems .
- MPPs are sometimes referred to as analytic databases because they are designed to allow for fast query processing and often have built-in analytic functions

- As the name implies, these database types process massive data sets in parallel across many processors and nodes
- An MPP architecture typically contains a single master node that is responsible for the coordination of all the data storage and processing across the cluster.
- It operates in a “shared-nothing” fashion, with each node containing local processing, memory, and storage and operating independently.

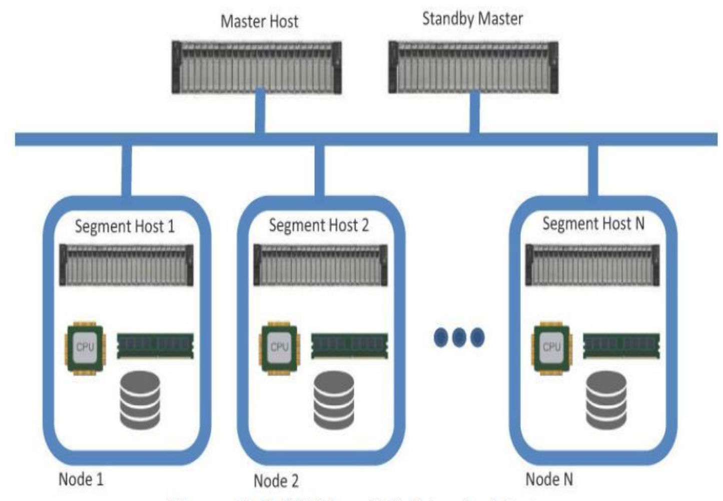


Figure 7-7 *MPP Shared-Nothing Architecture*

4. **Write a short note on edge streaming analytics.**

- One industry where data analytics is used extensively is the world of automobile racing.
- For example, in Formula One racing, each car has between 150 to 200 sensors that, combined, generate more than 1000 data points per second, resulting in hundreds of gigabytes of raw data per race.
- The sensor data is transmitted from the car and picked up by track-side wireless sensors. During a race, weather conditions may vary, tire conditions change, and accidents or other racing incidents almost always require an adaptable and flexible racing strategy.
- As the race develops, decisions such as when to pit, what tires to use, when to pass, and when to slow down all need to be made in seconds.
- Teams have found that enormous insights leading to better race results can be gained by analyzing data on the fly—and the data may come from many different sources, including trackside sensors, car telemetry, and weather reports.

Comparing Big Data and Edge Analytics

- From a business perspective, streaming analytics involves acting on data that is generated while it is still valuable, before it becomes stale.
- For example, roadway sensors combined with GPS way finding apps may tell a driver to avoid a certain highway due to traffic.
- This data is valuable for only a small window of time.
- Historically, it may be interesting to see how many traffic accidents or blockages have occurred on a certain segment of highway or to predict congestion based on past traffic data.
- However, for the driver in traffic receiving this information, if the data is not acted upon immediately, the data has little value.
- From a security perspective, having instantaneous access to analyzed and preprocessed data at the edge also allows an organization to realize anomalies in its network so those anomalies can be quickly contained before spreading to the rest of the network.
- To summarize, the key values of edge streaming analytics include the following:

  Reducing data at the edge:

  - The aggregate data generated by IoT devices is generally in proportion to the number of devices. The scale of these devices is likely to be huge, and so is the quantity of data they generate.
  - Passing all this data to the cloud is inefficient and is unnecessarily expensive in terms of bandwidth and network infrastructure.

Analysis and response at the edge:

- Some data is useful only at the edge (such as a factory control feedback system).
- In cases such as this, the data is best analyzed and acted upon where it is generated.

Time sensitivity:

- When timely response to data is required, passing data to the cloud for future processing results in unacceptable latency.
- Edge analytics allows immediate responses to changing conditions.

Edge Analytics Core Functions

- To perform analytics at the edge, data needs to be viewed as real-time flows.
- Whereas big data analytics is focused on large quantities of data at rest, edge analytics continually processes streaming flows of data in motion. Streaming analytics at the edge can be broken down into three simple stages:
- **Raw input data:**
  - ~ This is the raw data coming from the sensors into the analytics processing unit.
- **Analytics processing unit (APU):**
  - ~ The APU filters and combines data streams (or separates the streams, as necessary), organizes them by time windows, and performs various analytical functions.
  - ~ It is at this point that the results may be acted on by micro services running in the APU.
- **Output streams:**
  - ~ The data that is output is organized into insightful streams and is used to influence the behavior of smart objects, and passed on for storage and further processing in the cloud.
  - ~ Communication with the cloud often happens through a standard publisher/subscriber messaging protocol, such as MQTT.

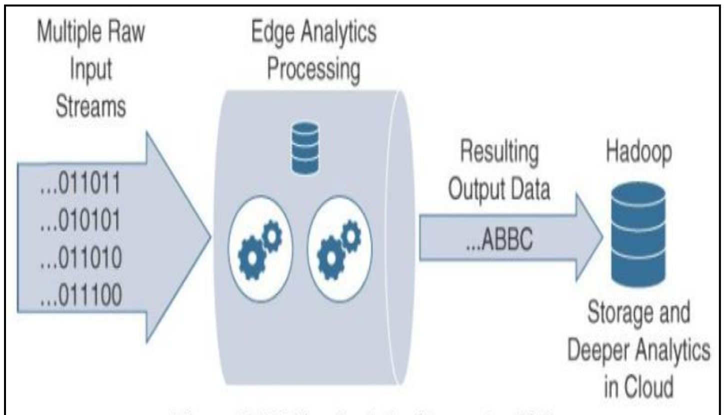

Multiple Raw Input Streams
...011011
...010101
...011010
...011100
Edge Analytics Processing
Resulting Output Data
...ABBC
Hadoop
Storage and Deeper Analytics in Cloud

Figure 7-12 *Edge Analytics Processing Unit*

5. **Explain OCTAVE AND FAIR in detail.**

- OCTAVE (Operationally Critical Threat, Asset and Vulnerability Evaluation) has undergone
- multiple iterations. The version this section focuses on is OCTAVE Allegro, which is intended to
- be a lightweight and less burdensome process to implement. Allegro assumes that a robust security
- team is not on standby or immediately at the ready to initiate a comprehensive security review.
- This approach and the assumptions it makes are quite appropriate, given that many operational
- technology areas are similarly lacking in security-focused human assets. illustrates the
- OCTAVE Allegro steps and phases.

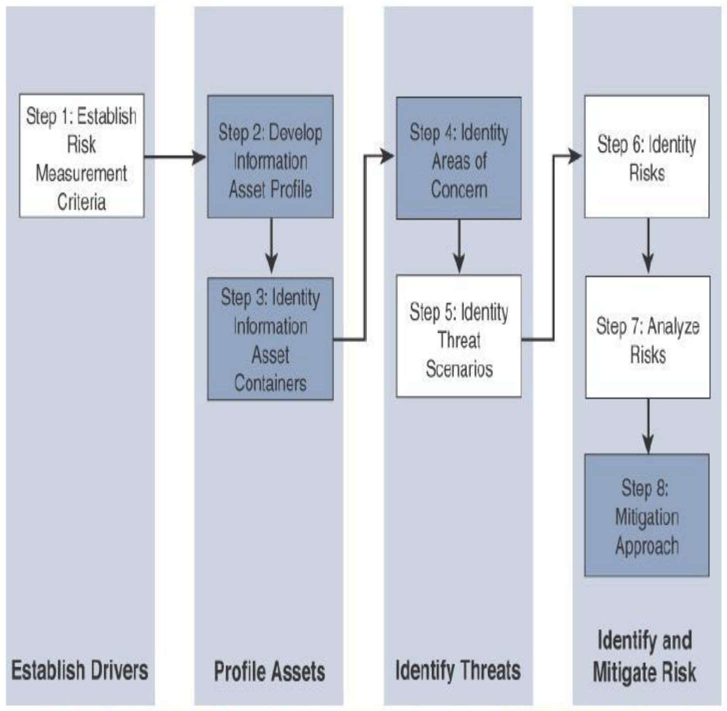

Step 1: Establish Risk Measurement Criteria
Step 2: Develop Information Asset Profile
Step 3: Identify Information Asset Containers
Step 4: Identify Areas of Concern
Step 5: Identify Threat Scenarios
Step 6: Identify Risks
Step 7: Analyze Risks
Step 8: Mitigation Approach
Establish Drivers
Profile Assets
Identify Threats
Identify and Mitigate Risk

Figure 8-5 *OCTAVE Allegro Steps and Phases (see https://blog.compass-*

- OCTAVE is a balanced information-focused process.
- What it offers in terms of discipline and largely unconstrained breadth, however, is offset by its lack of security specificity.
- There is an assumption that beyond these steps are seemingly means of identifying specific mitigations that can be mapped to the threats and risks exposed during the analysis process.

FAIR :

- FAIR (Factor Analysis of Information Risk) is a technical standard for risk definition from The Open Group.
- While information security is the focus, much as it is for OCTAVE, FAIR has clear applications within operational technology.
- Like OCTAVE, it also allows for non-malicious actors as a potential cause for harm, but it goes to greater lengths to emphasize the point. For many operational groups, it is a welcome acknowledgement of existing contingency planning.
- Unlike with OCTAVE, there is a significant emphasis on naming, with risk taxonomy definition as a very specific target.
- FAIR places emphasis on both unambiguous definitions and the idea that risk and associated attributes are measurable.
- Measurable, quantifiable metrics are a key area of emphasis, which should lend itself well to an operational world with a richness of operational data.
- At its base, FAIR has a definition of risk as the probable frequency and probable magnitude of loss.
- With this definition, a clear hierarchy of sub-elements emerges, with one side of the taxonomy focused on frequency and the other on magnitude.
- Loss even frequency is the result of a threat agent acting on an asset with a resulting loss to the organization.

4. Explain the Hadoop big data tool ?

Hadoop

- Hadoop is the most recent entrant into the data management market, but it is arguably the most popular choice as a data repository and processing engine.
- Hadoop was originally developed as a result of projects at Google and Yahoo!, and the original intent for Hadoop was to index millions of websites and quickly return search results for open source search engines.
- Initially, the project had two key elements:

- **Hadoop Distributed File System (HDFS):** A system for storing data across multiple nodes

MapReduce:

A distributed processing engine that splits a l arge task into smaller ones that can be run in parallel

- Much like the MPP and NoSQL systems discussed earlier, Hadoop relies on a scale-out architecture that leverages local processing, memory, and storage to distribute tasks and provide a scalable storage system for data.
- Both MapReduce and HDFS take advantage of this distributed architecture to store and process massive amounts of data and are thus able to leverage resources from all nodes in the cluster.
- For HDFS, this capability is handled by specialized nodes in the cluster, including NameNodes and DataNodes :

NameNodes:

- These are a critical piece in data adds, moves, deletes, and reads on HDFS.
- They coordinate where the data is stored, and maintain a map of where each block of data is stored and where it is replicated.
- All interaction with HDFS is coordinated through the primary (active) NameNode, with a secondary (standby) NameNode notified of the changes in the event of a failure of the primary.
- The NameNode takes write requests from clients and distributes those files across the available nodes in configurable block sizes, usually 64 MB or 128 MB blocks.
- The NameNode is also responsible for instructing the DataNodes where replication should occur.

DataNodes

- These are the servers where the data is stored at the direction of the NameNode.

- It is common to have many DataNodes in a Hadoop cluster to store the data.
- Data blocks are distributed across several nodes and often are replicated three, four, or more times across nodes for redundancy.
- Once data is written to one of the DataNodes, the DataNode selects two (or more) additional nodes, based on replication policies, to ensure data redundancy across the cluster.
- MapReduce leverages a similar model to batch process the data stored on the cluster nodes.
- Batch processing is the process of running a scheduled or ad hoc query across historical data stored in
- the HDFS. A query is broken down into smaller tasks and distributed across all the nodes running MapReduce in a cluster.
- While this is useful for understanding patterns and trending in historical sensor or machine data, it has one significant drawback: time

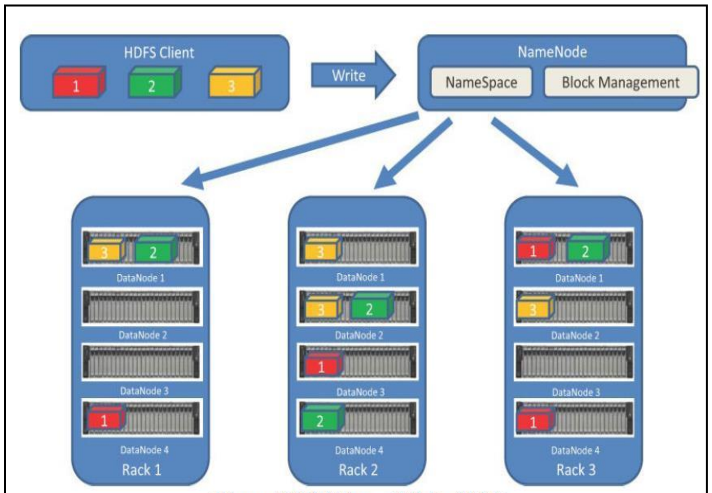


Figure 7-9 *Writing a File to HDFS*

YARN

- Introduced with version 2.0 of Hadoop, YARN (Yet Another Resource Negotiator) was designed to enhance the functionality of MapReduce.
- With the initial release, MapReduce was responsible for batch data processing and job tracking and resource management across the cluster.
- YARN was developed to take over the resource negotiation and job/task tracking, allowing MapReduce to be responsible only for data processing.

The Hadoop Ecosystem

- Since the initial release of Hadoop in 2011, many projects have been developed to add incremental functionality to Hadoop and have collectively become known as the Hadoop ecosystem.
  - ➢ Apache Kafka
  - ➢ Apache Spark
  - ➢ Apache Storm and Apache Flink
  - ➢ Lambda Architecture

Apache Kafka

- Part of processing real-time events, such as those commonly generated by smart objects, is having them ingested into a processing engine.
- The process of collecting data from a sensor or log file and preparing it to be processed and analyzed is typically handled by messaging systems.
- Messaging systems are designed to accept data, or messages, from where the data is generated and deliver the data to stream-processing engines such as Spark Streaming or Storm.
- Apache Kafka is a distributed publisher-subscriber messaging system that is built to be scalable and fast.
- It is composed of topics, or message brokers, where producers write data and consumers read data from these topics.
- the data flow from the smart objects (producers), through a topic in Kafka, to the real-time processing engine.
- Due to the distributed nature of Kafka, it can run in a clustered configuration that can handle many producers and consumers simultaneously and exchanges information between nodes, allowing topics to be distributed over multiple nodes.
- The goal of Kafka is to provide a simple way to connect to data sources and allow consumers to connect to that data in the way they would like.

Apache Spark

- Apache Spark is an in-memory distributed data analytics platform designed to accelerate processes in the Hadoop ecosystem.
- The "in-memory" characteristic of Spark is what enables it to run jobs very quickly.
- At each stage of a MapReduce operation, the data is read and written back to the disk, which means latency is introduced through each disk operation.
- However, with Spark, the processing of this data is moved into high-speed memory, which has significantly lower latency.
- This speeds the batch processing jobs and also allows for near-real-time processing of events.

Apache Storm and Apache Flink

- As you work with the Hadoop ecosystem, you will inevitably notice that different projects are very similar and often have significant overlap with other projects.
- This is the case with data streaming capabilities.
- For example, Apache Spark is often used for both distributed streaming analytics and batch processing.
- Apache Storm and Apache Flink are other Hadoop ecosystem projects designed for distributed stream processing and are commonly deployed for IoT use cases.
- Storm can pull data from Kafka and process it in a near-real-time fashion, and so can Apache Flink.
- This space is rapidly evolving, and projects will continue to gain and lose popularity as they evolve.

Lambda Architecture

- Ultimately the key elements of a data infrastructure to support many IoT use cases involves the collection, processing, and storage of data using multiple technologies.
- Querying both data in motion (streaming) and data at rest (batch processing) requires a combination of the Hadoop ecosystem projects discussed.
- One architecture that is currently being leveraged for this functionality is the Lambda Architecture.
- Lambda is a data management system that consists of two layers for ingesting data (Batch and Stream) and one layer for providing the combined data (Serving).
- These layers allow for the packages discussed previously, like Spark and MapReduce, to operate on the data independently, focusing on the key attributes for which they are designed and optimized. Data is taken from a message broker, commonly Kafka, and processed by each layer in parallel, and the resulting data is delivered to a data store where additional processing or queries can be run.
- This parallel data flow through the Lambda Architecture.
- The Lambda Architecture is not limited to the packages in the Hadoop ecosystem, but due to its breadth and flexibility, many of the packages in the ecosystem fill the requirements of each layer nicely:

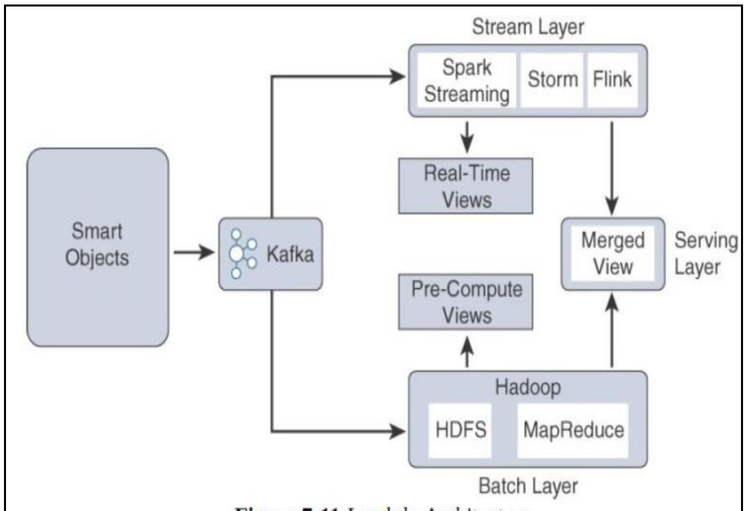


Figure 7-11 *Lambda Architecture*

Stream layer:

- This layer is responsible for near-real-time processing of events.
- Technologies such as Spark Streaming, Storm, or Flink are used to quickly ingest, process, and analyze data on this layer.
- Alerting and automated actions can be triggered on events that require rapid response

or could result in catastrophic outcomes if not handled immediately.

Batch layer:

- The Batch layer consists of a batch-processing engine and data store.
- If an organization is using other parts of the Hadoop ecosystem for the other layers, MapReduce and HDFS can easily fit the bill.
- Other database technologies, such as MPPs, NoSQL, or data warehouses, can also provide what is needed by this layer.

Serving layer:

- The Serving layer is a data store and mediator that decides which of the ingest layers to query based on the expected result or view into the data.
- If an aggregate or historical view is requested, it may invoke the Batch layer. If real-time analytics is needed, it may invoke the Stream layer.
- The Serving layer is often used by the data consumers to access both layers simultaneously.

5. What are the challenges in IoT security?

- The security challenges faced in IoT are by no means new and are not limited to specific industrial environments.
- The following sections discuss some of the common challenges faced in IoT.

Erosion of Network Architecture

- There is a wide variety in secured network designs within and across different industries.
- For example, power utilities have a strong history of leveraging modern technologies for operational activities, and in North America there are regulatory requirements in place from regulatory authorities, such as North American Electric Reliability Corporation's (NERC's) Critical Infrastructure Protection (CIP)

Pervasive Legacy Systems

- Due to the static nature and long lifecycles of equipment in industrial environments, many operational systems may be deemed legacy systems.
- For example, in a power utility environment, it is not uncommon to have racks of old mechanical equipment still operating alongside modern intelligent electronic devices (IEDs).
- In many cases, legacy components are not restricted to isolated network segments but have now been consolidated into the IT operational environment.
- From a security perspective, this is potentially dangerous as many devices may have historical vulnerabilities or weaknesses that have not been patched and updated, or it may be that patches are not even available due to the age of the equipment.

Insecure Operational Protocols

- The structure and operation of most of these protocols is often publicly available.
- While they may have been originated by a private firm, for the sake of interoperability, they are typically published for others to implement.
- Thus, it becomes a relatively simple matter to compromise the protocols themselves and introduce malicious actors that may use them to compromise control systems for either reconnaissance or attack purposes that could lead to undesirable impacts in normal system operation.

Device Insecurity

- Beyond the communications protocols that are used and the installation base of legacy systems, control and communication elements themselves have a history of vulnerabilities.
- To understand the nature of the device insecurity, it is important to review the history of what vulnerabilities were discovered and what types of devices were affected.
- A review of the time period 2000 to 2010 reveals that the bulk of discoveries were at the higher levels of the operational network, including control systems trusted to operate plants,

transmission systems, oil pipelines, or whatever critical function is in use.

6. Write a short note on network analytics.

- Another form of analytics that is extremely important in managing IoT systems is network-based analytics .
- Network analytics is concerned with discovering patterns in the communication flows from a network traffic perspective.
- Network analytics has the power to analyze details of communications patterns made by protocols and correlate this across the network.
- It allows you to understand what should be considered normal behavior in a network and to quickly identify anomalies that suggest network problems due to suboptimal paths, intrusive malware, or excessive congestion.

- Another form of analytics that is extremely important in managing IoT systems is network-based analytics .
- Network analytics is concerned with discovering patterns in the communication flows from a network traffic perspective.
- Network analytics has the power to analyze details of communications patterns made by protocols and correlate this across the network.
- It allows you to understand what should be considered normal behavior in a network and to quickly identify anomalies that suggest network problems due to suboptimal paths, intrusive malware, or excessive congestion.

7. What is NoSQL?

- NoSQL ("not only SQL") is a class of databases that support semi-structured and unstructured data, in addition to the structured data handled by data warehouses and MPPs.

- NoSQL is not a specific database technology; rather, it is an umbrella term that encompasses several different types of databases, including the following:

Document stores:

- ➢ This type of database stores semi-structured data, such as XML or JSON.
- ➢ Document stores generally have query engines and indexing features that allow for many optimized queries.

Key-value stores:

- ➢ This type of database stores associative arrays where a key is paired with an associated value.
- ➢ These databases are easy to build and easy to scale.

Wide-column stores:

- ➢ This type of database stores similar to a key-value store, but the formatting of the values can vary from row to row, even in the same table.

Graph stores:

- ➢ This type of database is organized based on the relationships between elements.
- ➢ Graph stores are commonly used for social media or natural language processing, where the connections between data are very relevant.
- ➢ NoSQL was developed to support the high-velocity, urgent data requirements of modern web applications that typically do not require much repeated use.
- ➢ The original intent was to quickly ingest rapidly changing server logs and clickstream data generated by web-scale applications that did not neatly fit into the rows and columns required by relational databases.
- ➢ Similar to other data stores, like MPPs and Hadoop (discussed later), NoSQL is built to scale horizontally, allowing the database to span multiple hosts, and can even be distributed geographically.
- ➢ Expanding NoSQL databases to other nodes is similar to expansion in other distributed data systems, where additional hosts are managed by a master node or process.
- ➢ This expansion can be automated by some NoSQL implementations or can be provisioned manually.

- This level of flexibility makes NoSQL a good candidate for holding machine and sensor data associated with smart objects

8. What is Machine Learning?

- ML is central to IoT.
- Data collected by smart objects needs to be analyzed, and intelligent actions need to be taken based on these analyses.
- Performing this kind of operation manually is almost impossible (or very, very slow and inefficient).
- Machines are needed to process information fast and react instantly when thresholds are met
- Ex: advances in self-driving vehicle--abnormal pattrn recognition in a crowd and automated intelligent and machine-assisted decision systemMachine learning is, in fact, part of a larger set of technologies commonly grouped under the term artificial intelligence (AI).
- AI includes any technology that allows a computing system to mimic human intelligence using any technique, from very advanced logic to basic “if-then-else” decision loops.
- Any computer that uses rules to make decisions belong to this group.
- ML is concerned with any process where the computer needs to receive a set of data that is processed to help perform a task with more efficiency.
- ML is a vast field but can be simply divided in two main categories: supervised and unsupervised learning

9. Dstinguish between supervised learning and unsupervised learning?

Parameters	Supervised learning	Unsupervised learning
Process	In a supervised learning model, input and output variables will be given.	In unsupervised learning model, only input data will be given
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data which is not labeled
Algorithms Used	Support vector machine, Neural network, Linear and logistics regression, random forest, and Classification trees.	Unsupervised algorithms can be divided into different categories: like Cluster algorithms, K-means, Hierarchical clustering, etc.
Computational Complexity	Supervised learning is a simpler method.	Unsupervised learning is computationally complex
Use of Data	Supervised learning model uses training data to learn a link between the input and the outputs.	Unsupervised learning does not use output data.
Computational Complexity	Supervised learning is a simpler method.	Unsupervised learning is computationally complex
Use of Data	Supervised learning model uses training data to learn a link between the input and the outputs.	Unsupervised learning does not use output data.
Accuracy of Results	Highly accurate and trustworthy method.	Less accurate and trustworthy method.
Real Time Learning	Learning method takes place offline.	Learning method takes place in real time.
Accuracy of Results	Highly accurate and trust worthy method.	Less accurate and trustworthy method.
Real Time Learning	Learning method takes place offline.	Learning method takes place in real time.

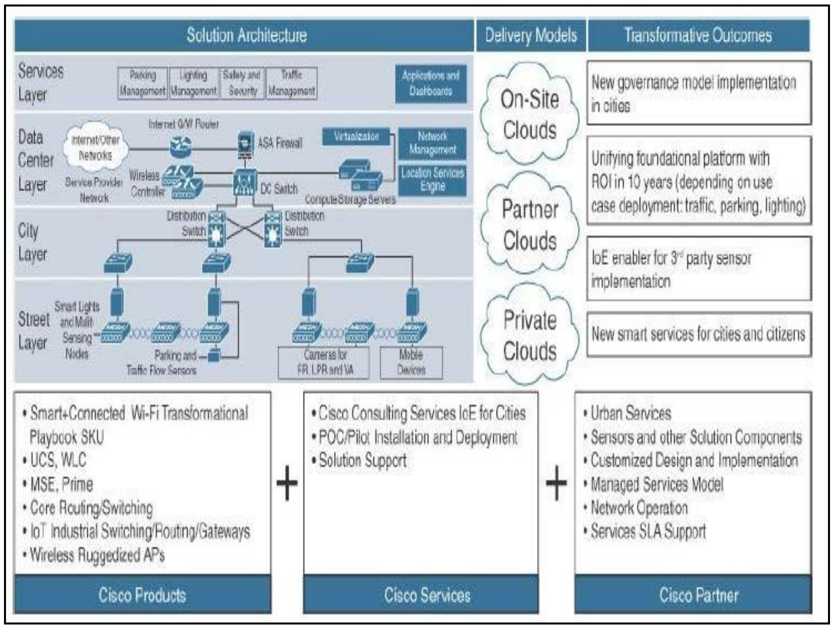
10. Compare b/n structured and unstructured data ?

Properties	Structured data	Unstructured data
Technology	It is based on Relational database table	It is based on character and binary data
Transaction management	Matured transaction and various concurrency techniques	No transaction management and no concurrency
Version management	Versioning over tuples, row, tables	Versioned as a whole
Flexibility	It is schema dependent and less flexible	It is more flexible and there is absence of schema
Scalability	It is very difficult to scale DB schema	It is more scalable.
Robustness	Very robust	Less robust
Query performance	Structured query allow complex joining	Only textual queries are possible
Format	It has a predefined format.	It has a variety of formats, i.e., it comes in a variety of shapes and sizes.
Analysis	It is easy to search	Searching for unstructured data is more difficult.

UNIT 5: IoT Physical Devices and Endpoints

1. Explain the smart city IoT architecture.

- A smart city IoT infrastructure is a four-layered architecture,
- Data flows from devices at the street layer to the city network layer and connect to the
- data center layer, where the data is aggregated, normalized, and virtualized.
- The data center layer provides information to the services layer, which consists of the
- applications that provide services to the city

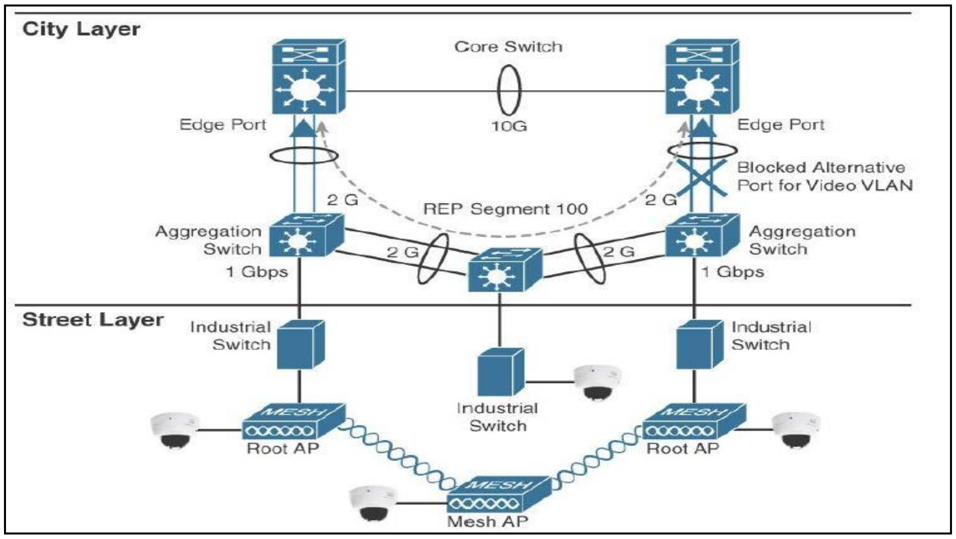


1) Street Layer

- The street layer is composed of devices and sensors that collect data and take action based on instructions from the overall solution, as well as the networking components needed to aggregate and collect data.
- A sensor is a data source that generates data required to understand the physical world. Sensor devices are able to detect and measure events in the physical world.
- ICT (information and communication technology) connectivity solutions rely on sensors to collect the data from the world around them so that it can be analyzed and used to operationalize use cases for cities.
- A variety of sensors are used at the street layer for a variety of smart city use cases
- A magnetic sensor can detect a parking event by analyzing changes in the surrounding magnetic field when a heavy metal object, such as a car or a truck, comes close to it (or on top of it).
- A lighting controller can dim and brighten a light based on a combination of time based and ambient conditions. Video cameras combined with video analytics can detect vehicles, faces, and traffic conditions for various traffic and security use cases.
- An air quality sensor can detect and measure gas and particulate matter concentrations to give a hyper localized perspective on pollution in a given area.
- Device counters give an estimate of the number of devices in the area, which provides a rough idea of the number of vehicles moving or parked in a street or a public parking area, of pedestrians on a sidewalk, or even of birds in public parks or on public monuments—for cities where bird control has become an issue.

2) City Layer

- At the city layer, which is above the street layer, network routers and switches must be deployed to match the size of city data that needs to be transported.
- This layer aggregates all data collected by sensors and the end-node network into a single transport network.
- The city layer may appear to be a simple transport layer between the edge devices and the data center or the Internet.
- However, one key consideration of the city layer is that it needs to transport multiple types of protocols, for multiple types of IoT applications.
- Some applications are delay- and jitter-sensitive, and some other applications require a deterministic approach to frame delivery.
- As a result, the city layer must be built around resiliency, to ensure that a packet coming from a sensor or a gateway will always be forwarded successfully to the headend station.



igure 5.13 shows a common way of achieving this goal.

3) Data Center Layer

- Ultimately, data collected from the sensors is sent to a data center, where it can be processed and correlated.
- Based on this processing of data, meaningful information and trends can be derived, and information can be provided back.
- For example, an application in a data center can provide a global view of the city traffic and help authorities decide on the need for more or less common transport vehicles.
- At the same time, an automated response can be generated.
- For example, the same traffic information can be processed to automatically regulate and coordinate the street light durations at the scale of the entire city to limit traffic congestion.
- The key technology in creating any comprehensive smart solution with services is the cloud.
- With a cloud infrastructure, data is not stored in a data center owned directly or indirectly by city authorities.
- Instead, data is stored in rented logical containers accessed through the Internet.
- Because the containers can be extended or reduced based on needs, the storage size and computing power are flexible and can adapt to changing requirements or budget conditions.
- In addition, multiple contractors can store and process data at the same time, without the complexity of exclusively owned space.

- This proximity and flexibility also facilitate the exchange of information between smart systems and allow for the deployment of new applications that can leverage information from several IoT systems.
- Fig shows the vision of utilizing the cloud in smart solutions for cities.

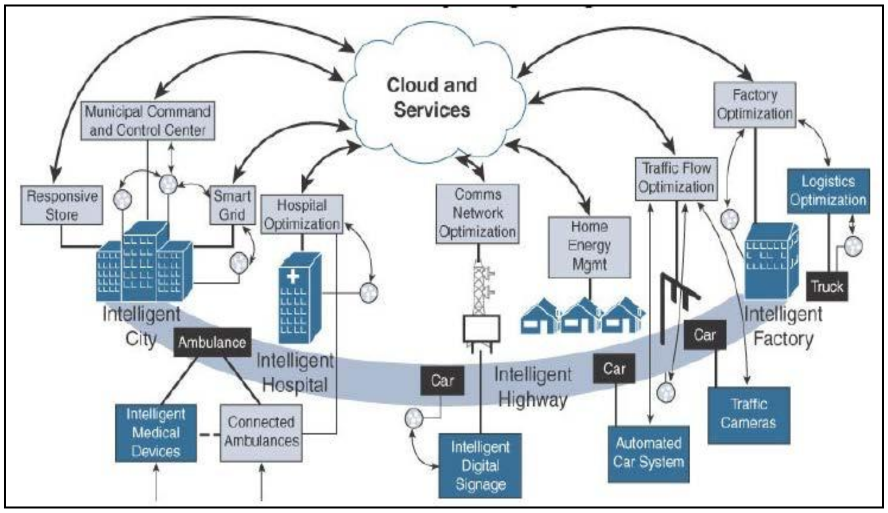


Figure 5.14 The Role of the Cloud for Smart City Application

- The cloud provides a scalable, secure, and reliable data processing engine that can handle the immense amount of data passing through it.
- Smart city issues require not just efficient use of infrastructure, which the cloud helps enable, they also require new data processing and management models.
- For example, cloud services allow for Software as a Service (SaaS) models that create cyclical returns on investment
- With the cloud approach shown in Figure 5.14 , smart cities can also take advantage of operating expense–based consumption models to overcome any financial hurdles in adopting solutions to their most critical issues.
- Critical data, such as air condition (humidity, temperature, pollution) levels monitoring, can be processed initially.
- Then, as the efficiency of IoT is scaled up, richer data processing can be enabled in the cloud applications.
- For example, the humidity level can be used to regulate the color and luminosity of street lights. In times when city budgets are strained, data processing can be scaled down to essential services.

4) Services Layer

- Ultimately, the true value of ICT connectivity comes from the services that the measured data can provide to different users operating within a city.
- Smart city applications can provide value to and visibility for a variety of user types, including city operators, citizens, and law enforcement.
- The collected data should be visualized according to the specific needs of each consumer of that data and the particular user experience requirements and individual use cases.
- For example, parking data indicating which spots are and aren’t currently occupied can drive a citizen parking app with a map of available spots, as well as an enforcement officer’s understanding of the state (utilization and payment) of the public parking space,

- while at the same time helping the city operator's perspective on parking problem areas in the city at any given time.
- With different levels of granularity and scale, the same data performs three different functions for three different users.
- Along the same lines, traffic information can be used by individual car drivers to find the least congested route.
- A variation of the same information can be made available to public transportation users to estimate travel times.
- Public transportation systems, such as buses, can be rerouted around known congestion points.
- The number of subway trains can be increased dynamically to respond to an increase in traffic congestion, anticipating the decisions of thousands or even millions of commuters to take public transportation instead of cars on days when roads are very congested.
- Here again, the same type of data is utilized by different types of users in different ways based on their specific use cases.

2. Explain the features of Arduino?

- Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software.
- It consists of a circuit board, which can be programed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are −

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.