SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – I – IoT Architecture and its Protocols– SCSA1408

UNIT 1 INTRODUCTION TO IoT

Introduction - IoT and digitization IoT impact Convergence of Information Technology and Operational Technology Ancestors without IP IoT enabled applications - IoT challenges

Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established. Over 9 billion 'Things' (physical objects) are currently connected to the Internet, as of now. In the near future, this number is expected to rise to a whopping 20 billion.

There are four main components used in IoT:

Low-power embedded systems: Less battery consumption, high performance are the inverse factors that play a significant role during the design of electronic systems.

Cloud computing: Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.

Availability of big data: We know that IoT relies heavily on sensors, especially in real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

Networking connection: In order to communicate, internet connectivity is a must where each physical object is represented by an IP address. However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.

There are two ways of building IoT:

- Form a separate internetwork including only physical objects.
- Make the Internet ever more expansive, but this requires hard-core technologies such as rigorous cloud computing and rapid big data storage

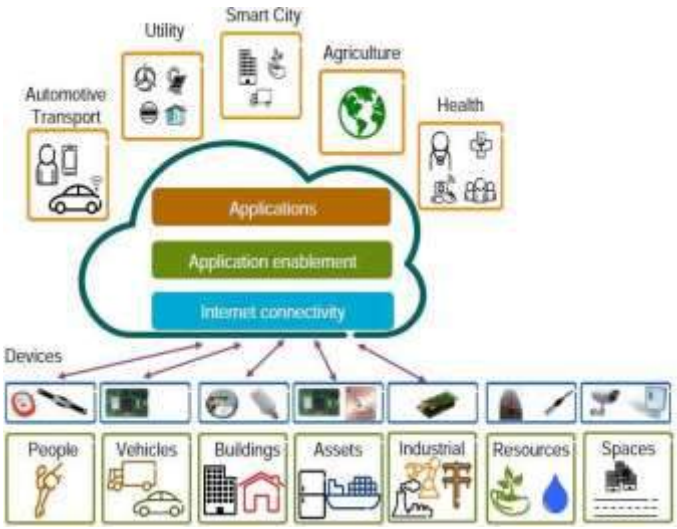


Fig. 1.1 IoT

IoT

- The IoT is a widely used term for a set of technologies, systems, and design principles associated with the emerging wave of Internet-connected things that are based on the physical environment.
- In many respects, it can initially look the same as M2M communication connecting sensors and other devices to Information and Communication Technology (ICT)systems via wired or wireless networks.
- In contrast to M2M, however, IoT also refers to the connection of such systems and sensors to the broader Internet, as well as the use of general Internet technologies.
- In the longer term, it is envisaged that an IoT ecosystem will emerge not dissimilar totoday's Internet, allowing things and real world objects to connect, communicate, and interact with one another in the same way humans do via the web today.
- No longer will the Internet be only about people, media, and content, but it will also include all real-world assets as intelligent creatures exchanging information, interacting with people, supporting business processes of enterprises, and creatingknowledge.
- The IoT is not a new Internet, it is an extension to the existing Internet. IoT is about the technology, the remote monitoring, and control, and also about where thesetechnologies are applied. IoT can have a focus on the open innovative promises of the technologies at play, and also on advanced and complex processing inside veryconfined and close environments such as
- Looking towards the applications and services in the IoT, we see that the application opportunities are open-ended, and only imagination will set the limit of what isachievable.
- Starting from typical M2M applications, one can see application domains emerging that are driven from very diverse needs from across industry, society, and people, and can be of both local interest and global interest.
- Applications can focus on safety, convenience, or cost reduction, optimizing business processes, or fulfilling various requirements on sustainability and assisted living.

- Listing all possible application segments is futile, as is providing a ranking of themost important ones. We can point to examples of emerging application domains that are driven by different trends and interests .
- As can be seen, they are very diverse and can include applications like urban agriculture, robots and food safety tracing, and we will give brief explanations of what these three examples might look like.

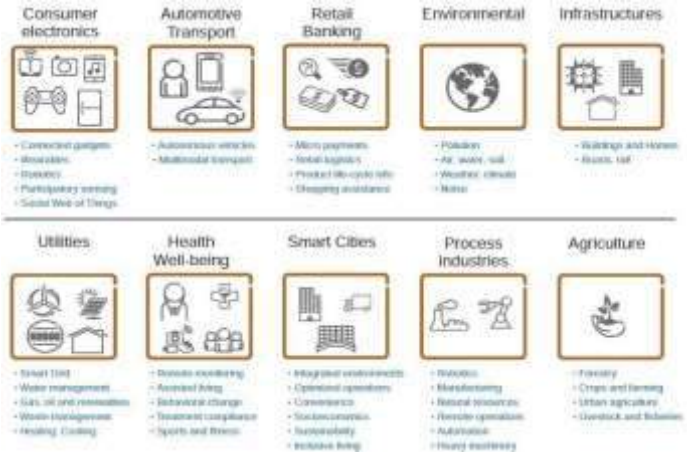


Fig 1.2 . Emerging IoT Applications

- **Urban Agriculture.** Already today, more than 50% of the world's population lives in urban areas and cities. The increased attention on sustainable living includes reducing transportation, and in the case of food production, reducing the needs for pesticides.
- The prospect of producing food at the place where it is consumed (i.e. in urban areas) is a promising example. By using IoT technologies, urban agriculture could be highly optimized.
- Sensors and actuators can monitor and control the plant environment and tailor the conditions according to the needs of the specific specimen.
- Water supply through a combination of rain collection and remote feeds can becombined on demand. City or urban districts can have separate infrastructures for the provisioning of different fertilizers.
- Weather and light can be monitored, and necessary blinds that can shield and protect, as well as create greenhouse microclimates, can be automatically controlled.
- Fresh air generated by plants can be collected and fed into buildings, and tanks of algae that consume waste can generatefertilizers.
- **Robots.** The mining industry is undergoing a change for the future. Production rates must be increased, cost per produced unit decreased, and the lifetime of mines and sites must be prolonged.
- In addition, human workforce safety must be higher, with fewer or no accidents, and environmental impact must be decreased by reducing energy consumption and carbonemissions.
- The mining industry answer to this is to turn each mineinto a fully automated and controlled operation. The process chain of the mine involving blasting, crushing, grinding, and ore processing will be highly automated and interconnected.
- The heavy machinery used will be remotely controlled and monitored, mine sites will be connected, and shafts monitored in terms of air and gases.

- Sensors in the mine can provide information about the location of the machines.
- The trend is also that local control rooms will be replaced by larger control rooms at the corporate headquarters. Sensors and actuators to remotely control both the sites and the massive robots in terms of mining machines for drilling, haulage, and processing are the instruments to make this happen.
- **Food Safety.** After several outbreaks of food-related illnesses in the U.S., the U.S. Food and Drug Administration (USFDA) created its Food Safety and Modernization Act (FSMA 2011).
- The main objective with FSMA is to ensure that the U.S. food supply is safe. Similar food safety objectives have also been declared by the European Union and the Chinese authorities.
- These objectives will have an impact across the entire food supply chain, from the farm to the table, and require a number of actors to integrate various parts of their businesses.
- From the monitoring of farming conditions for plant and animal health, registration of the use of pesticides and animal food, the logistics chain to monitor environmental conditions as produce is being transported, and retailers handling of food all will be connected.
- Sensors will provide the necessary monitoring capabilities, and tags like radio frequency identification (RFID) will be used to identify the items so they can betracked and traced throughout the supply chain. The origin of food can also becompletely transparent to the consumers.
- As can be seen by these very few examples, IoT can target very point and closed domain-oriented applications, as well as very open and innovation driven applications.
- Applications can stretch across an entire value chain and provide lifecycle perspectives. Applications can be for businessto-business (B2B) as well as for business-to-consumer (B2C), and can be complex and involve numerous actors, as well as large sets of heterogeneous data sources.

IoT and Digitization

- IoT and digitization are terms that are often used interchangeably. In most contexts, this duality is fine, but there are key differences to be aware of.
- At a high level, IoT focuses on connecting "things," such as objects and machines, to a computer network, such as the Internet. IoT is a well-understood term used across the industry as a whole. On the other hand, digitization can mean different things to different people but generally encompasses the connection of "things" with the data they generate and the business insights that result.
- For example, in a shopping mall where Wi-Fi location tracking has been deployed, the "things" are the Wi-Fi devices. Wi-Fi location tracking is simply the capability of knowing where a consumer is in a retail environment through his or her smart phone's connection to the retailer's Wi-Fi network. While the value of connecting Wi-Fi devices or "things" to the Internet is obvious and appreciated by shoppers, tracking real-time location of
- Wi-Fi clients provides a specific business benefit to the mall and shop owners. In this case, it helps the business understand where shoppers tend to congregate and how much time they spend in different parts of a mall or store. Analysis of this data can lead to sig- nificant changes to the locations of product displays and advertising, where to place cer- tain types of shops, how much rent to charge, and even where to station security guards.
- Digitization, as defined in its simplest form, is the conversion of information into a digital format. Digitization has been happening in one form or another for several decades.

- For example, the whole photography industry has been digitized. Pretty much everyone has digital cameras these days, either standalone devices or built into their mobile phones. Almost no one buys film and takes it to a retailer to get it developed. The digitization of photography has completely changed our experience when it comes to capturing images.
- Other examples of digitization include the video rental industry and transportation. In the past, people went to a store to rent or purchase videotapes or DVDs of movies.
- With digitization, just about everyone is streaming video content or purchasing movies as downloadable files.
- The transportation industry is currently undergoing digitization in the area of taxi services. Businesses such as Uber and Lyft use digital technologies to allow people to get a ride using a mobile phone app. This app identifies the car, the driver, and the fare.
- The rider then pays the fare by using the app. This digitization is a major disruptive force to companies providing traditional taxi services.
- In the context of IoT, digitization brings together things, data, and business process to make networked connections more relevant and valuable. A good example of this that many people can relate to is in the area of home automation with popular products, such as Nest. With Nest, sensors determine your desired climate settings and also tie in other smart objects, such as smoke alarms, video cameras, and various third-party devices.
- In the past, these devices and the functions they perform were managed and controlled separately and could not provide the holistic experience that is now possible. Nest is just one example of digitization and IoT increasing the relevancy and value of networked, intelligent connections and making a positive impact on our lives.
- Companies today look at digitization as a differentiator for their businesses, and IoT is a prime enabler of digitization. Smart objects and increased connectivity drive digitization, and this is one of the main reasons that many companies, countries, and governments are embracing this growing trend.

IoT Applications

The IoT can find its applications in almost every aspect of our daily life. Below are some of the examples.

1) Prediction of natural disasters: The combination of sensors and their autonomous coordination and simulation will help to predict the occurrence of land-slides or other natural disasters and to take appropriate actions in advance.

2) Industry applications: The IoT can find applications in industry e.g., managing a fleet of cars for an organization. The IoT helps to monitor their environmental performance and process the data to determine and pick the one that need maintenance.

3) Water Scarcity monitoring: The IoT can help to detect the water scarcity at different places. The networks of sensors, tied together with the relevant simulation activities might not only monitor long term water interventions such as catchment area management, but may even be used to

alert users of astream, for instance, if an upstream event, such as the accidental release of sewage into the stream, might have dangerous implications.

4) Design of smart homes: The IoT can help in the design of smart homes e.g., energy consumption management, interaction with appliances, detecting emergencies, home safety and finding things easily, home security etc.

5) Medical applications: The IoT can also find applications in medical sectorfor saving lives or improving the quality of life e.g., monitoring health parameters, monitoring activities, support for independent living, m The IoT can find its applications in almost every aspect of our daily life. Below are some of the examples.

6) Prediction of natural disasters: The combination of sensors and their autonomous coordination and simulation will help to predict the occurrence of land-slides or other natural disasters and to take appropriate actions in advance.

7) Industry applications: The IoT can find applications in industry e.g., managing a fleet of cars for an organization. The IoT helps to monitor their environmental performance and process the data to determine and pick the one that need maintenance.

8) Water Scarcity monitoring: The IoT can help to detect the water scarcity at different places. The networks of sensors, tied together with the relevant simulation activities might not only monitor long term water interventions such as catchment area management, but may even be used to alert users of a stream, for instance, if an upstream event, such as the accidental release of sewage into the stream, might have dangerous implications.

9) Design of smart homes: The IoT can help in the design of smart homes e.g., energy consumption management, interaction with appliances, detecting emergencies, home safety and finding things easily, home security etc.

10) Medical applications: The IoT can also find applications in medical sector for saving lives or improving the quality of life e.g., monitoring health parameters, monitoring activities, support for independent living, monitoring medicines intake etc.

11) Agriculture application: A network of different sensors can sense data, perform data processing and inform the farmer through communication infrastructure e.g., mobile phone text message about the portion of land that need particular attention. This may include smart packaging of seeds, fertilizer and pest control mechanisms that respond to specific local conditions and indicate actions. Intelligent farming system will help agronomists to have better understanding of the plant growth models and to have efficient farming practices by having the knowledge of land conditions and climate variability. This will significantly increase the agricultural productivity by avoiding the inappropriate farming

conditions.

12) Intelligent transport system design: The Intelligent transportation system will provide efficient transportation control and management using advanced technology of sensors, information and network. The intelligent transportation can have many interesting featuressuch as non-stop electronic highway toll, mobile emergency command and scheduling, transportation law enforcement, vehicle rules violation monitoring, reducingenvironmental pollution, anti-theft system, avoiding traffic jams, reporting traffic incidents, smart beaconing, minimizing arrival delays etc.

13) Design of smart cities: The IoT can help to design smart cities e.g., monitoring air quality, discovering emergency routes, efficient lighting up of the city, watering gardens etc.

14) Smart metering and monitoring: The IoT design for smart metering and monitoring willhelp to get accurate automated meter reading and issuance of invoice to the customers. The IoT can be used to design such scheme for wind turbine maintenance and remote monitoring, gas, water as well as environmental metering and monitoring.

15) Smart Security: The IoT can also find applications in the field of security and surveillance e.g., surveillance of spaces, tracking of people and assets, infrastructure and equipment maintenance, alarming, monitoringmedicines intake etc.

16) Agriculture application: A network of different sensors can sense data, perform data processing and inform the farmer through communicationinfrastructure e.g., mobile phone text message about the portion of land that need particular attention. This may include smart packaging of seeds, fertilizer and pest control mechanisms that respond to specific local conditions and indicate actions. Intelligent farming system will help agronomists to have better understanding of the plant growth models and to have efficient farming practices by having the knowledge of land conditions and climate variability. This will significantly increase the agricultural productivity by avoiding the inappropriate farming conditions.

17) Intelligent transport system design: The Intelligent transportation system will provide efficient transportation control and management using advanced technology of sensors, information and network. The intelligent transportation can have many interesting features such as non-stop electronic highway toll, mobile emergency command and scheduling, transportation law enforcement, vehicle rules violation monitoring, reducing environmental pollution, anti-theft system, avoiding traffic jams, reporting traffic incidents, smart beaconing, minimizing arrival delays etc.

18) Design of smart cities: The IoT can help to design smart cities e.g., monitoring air quality, discovering emergency routes, efficient lighting up ofthe city, watering gardens etc.

19) Smart metering and monitoring: The IoT design for smart metering and monitoring will

help to get accurate automated meter reading and issuance of invoice to the customers. The IoT can be used to design such scheme for wind turbine maintenance and remote monitoring, gas, water as well as environmental metering and monitoring.

20) Smart Security: The IoT can also find applications in the field of security andsurveillance e.g., surveillance of spaces, tracking of people and assets, infrastructure and equipment maintenance, alarming etc.

Connected Roadways

People have been fantasizing about the self-driving car, or autonomous vehicle, in literature and film for decades. While this fantasy is now becoming a reality with well-known projects like Google's self-driving car, IoT is also a necessary component for implementing a fully connected transportation infrastructure. IoT is going to allow self-driving vehicles to better interact with the transportation system around them through bidirectional data exchanges while also providing important data to the riders. Self-driving vehicles need always-on, reliable communications and data from other transportation-related sensors to reach their full potential. Connected roadways is the term associated with both the driver and driverless cars fully integrating with the surrounding transportation infrastructure. Figure shows a self-driving car designed by Google.

Fig 1.3 : Self driving car

Basic sensors reside in cars already. They monitor oil pressure, tire pressure, temperature, and other operating conditions, and provide data around the core car functions. From behind the steering wheel, the driver can access this data while also controlling the car using equipment such as a steering wheel, pedals, and so on. The need for all this sensory information and control is obvious. The driver must be able to understand, handle, and make critical decisions while concentrating on driving safely. The Internet of Things is replicating this concept on a much larger scale.

Today, we are seeing automobiles produced with thousands of sensors, to measure every- thing from fuel consumption to location to the entertainment your family is watching during the ride. As automobile manufacturers strive to reinvent the driving experience, these sensors are becoming IP-enabled to allow easy communication with other systems both inside and outside the car. In addition, new sensors and communication technologies are being developed to allow vehicles to "talk" to other vehicles, traffic signals, school zones, and other elements of the transportation infrastructure. We are now starting to realize a truly connected transportation solution.

Connected Factory

For years, traditional factories have been operating at a disadvantage, impeded by production environments that are “disconnected” or, at the very least, “strictly gated” to corporate business systems, supply chains, and customers and partners. Managers of these traditional factories are essentially “flying blind” and lack visibility into their operations. These operations are composed of plant floors, front offices, and suppliers For years, traditional factories have been operating at a disadvantage, impeded by production environments that are “disconnected” or, at the very least, “strictly gated” to corporate business systems, supply chains, and customers and partners. Managers of these traditional factories are essentially “flying blind” and lack visibility into their operations. These operations are composed of plant floors, front offices, and suppliers operating in independent silos. Consequently, rectifying downtime issues, quality problems, and the root causes of various manufacturing inefficiencies is often difficult.

The main challenges facing manufacturing in a factory environment today include the following:

■ Accelerating new product and service introductions to meet customer and market opportunities
■ Increasing plant production, quality, and uptime while decreasing cost
■ Mitigating unplanned downtime (which wastes, on average, at least 5% of production)
■ Securing factories from cyber threats
■ Decreasing high cabling and re-cabling costs (up to 60% of deployment costs)
■ Improving worker productivity and safety4

Adding another level of complication to these challenges is the fact that they often need to be addressed at various levels of the manufacturing business. For example, executive management is looking for new ways to manufacture in a more cost-effective manner while balancing the rising energy and material costs. Product development has time to market as the top priority. Plant managers are entirely focused on gains in plant efficiency and operational agility. The controls and automation department looks after the plant networks, controls, and applications and therefore requires complete visibility into all these systems.

Industrial enterprises around the world are retooling their factories with advanced technologies and architectures to resolve these problems and boost manufacturing flex- ibility and speed. These improvements help them achieve new levels of overall equipment effectiveness, supply chain responsiveness, and customer satisfaction. A convergence of factory-based operational technologies and architectures with global IT networks is starting to occur, and this is referred to as the connected factory.

As with the IoT solutions for the connected roadways previously discussed, there are already large numbers of basic sensors on factory floors. However, with IoT, these sensors not only become more advanced but also attain a new level of connectivity. They are smarter and gain the ability to communicate, mainly using the Internet Protocol (IP) over an Ethernet infrastructure.

In addition to sensors, the devices on the plant floor are becoming smarter in their ability to transmit and receive large quantities of real-time informational and diagnostic data. Ethernet connectivity is becoming pervasive and spreading beyond just the main con- trollers in a factory to devices such as the robots on the plant floor. In addition, more IP-enabled devices, including video cameras, diagnostic smart objects, and even personal mobile devices, are being added to the manufacturing environment.

For example, a smelting facility extracts metals from their ores. The facility uses both heat and chemicals to decompose the ore, leaving behind the base metal. This is a multistage process, and the data and controls are all accessed via various control rooms in a facility. Operators must go to a control room that is often hundreds of meters away for data and production changes. Hours of operator time are often lost to the multiple trips to the control room needed during a shift. With IoT and a connected factory solution, true “machine-to-people” connections are implemented to bring sensor data directly to operators on the floor via mobile devices. Time is no longer wasted moving back and forth between the control rooms and the plant floor. In addition, because the operators now receive data in real time, decisions can be made immediately to improve production and fix any quality problems. Another

example of a connected factory solution involves a real-time location system (RTLS). An RTLS utilizes small and easily deployed Wi-Fi RFID tags that attach to virtually any material and provide real-time location and status. These tags enable a facility to track production as it happens. These IoT sensors allow components and materials on an assembly line to "talk" to the network. If each assembly line's output is tracked in real time, decisions can be made to speed up or slow production to meet targets, and it is easy to determine how quickly employees are completing the various stages of production. Bottlenecks at any point in production and quality problems are also quickly identified.

Smart Connected Buildings

Another place IoT is making a disruptive impact is in the smart connected buildings space. In the past several decades, buildings have become increasingly complex, with systems overlaid one upon another, resulting in complex intersections of structural, mechanical, electrical, and IT components. Over time, these operational networks that support the building environment have matured into sophisticated systems; however, for the most part, they are deployed and managed as separate systems that have little to no interaction with each other.

The function of a building is to provide a work environment that keeps the workers comfortable, efficient, and safe. Work areas need to be well lit and kept at a comfortable temperature. To keep workers safe, the fire alarm and suppression system needs to be carefully managed, as do the door and physical security alarm systems. While intelligent systems for modern buildings are being deployed and improved for each of these functions, most of these systems currently run independently of each other—and they rarely take into account where the occupants of the building actually are and how many of them are present in different parts of the building. However, many buildings are beginning to deploy sensors throughout the building to detect occupancy. These tend to be motion sensors or sensors tied to video cameras. Motion detection occupancy sensors work great if everyone is moving around in a crowded room and can automatically shut the lights off when everyone has left, but what if a person in the room is out of sight of the sensor. It is a frustrating matter to be at the mercy of an unintelligent sensor on the wall that wants to turn off the lights on you.

Similarly, sensors are often used to control the heating, ventilation, and air-conditioning (HVAC) system. Temperature sensors are spread throughout the building and are used to influence the building management system's (BMS's) control of air flow into a room.

Another interesting aspect of the smart building is that it makes them easier and cheaper to manage. Considering the massive costs involved in operating such complex structures, not to mention how many people spend their working lives inside a building, managers have become increasingly interested in ways to make buildings more efficient and cheaper to manage. Have you ever heard people complain that they had too little working space
in their office, or that the office space wasn't being used efficiently? When people go to their managers and ask for a change to the floor plan, such as asking for an increase in the amount of space they work in, they are often asked to prove their case. But workplace floor efficiency and usage evidence tend to be anecdotal at best. When smart building sensors and occupancy detection are combined with the power of data analytics it becomes easy to demonstrate floor plan usage and prove your case. Alternatively, the building manager can use a similar approach to see where the floor is not being used efficiently and use this information to optimize the available space. This has brought about the age of building automation, empowered by IoT. Another promising IoT technology in the smart connected building, and one that is seeing widespread adoption, is the "digital ceiling." The digital ceiling is more than just a lighting control system. This technology encompasses several of the building's different networks—including lighting, HVAC, blinds, CCTV (closed-circuit television), and security systems—and combines them into a single IP network.

Smart Creatures

When you think about IoT, you probably picture only inanimate objects and machines being

connected. However, IoT also provides the ability to connect living things to the Internet. Sensors can be placed on animals and even insects just as easily as on machines, and the benefits can be just as impressive.

One of the most well-known applications of IoT with respect to animals focuses on what is often referred to as the "connected cow." Sparked, a Dutch company, developed a sen- sor that is placed in a cow's ear. The sensor monitors various health aspects of the cow as well as its location and transmits the data wirelessly for analysis by the farmer.

The data from each of these sensors is approximately 200 MB per year, and you obvi- ously need a network infrastructure to make the connection with the sensors and store the information. Once the data is being collected, however, you get a complete view of the herd, with statistics on every cow. You can learn how environmental factors may be affecting the herd as a whole and about changes in diet. This enables early detection of disease as cows tend to eat less days before they show symptoms. These sensors even allow the detection of pregnancy in cows.

Another application of IoT to organisms involves the placement of sensors on roaches. While the topic of roaches is a little unsettling to many folks, the potential benefits of IoT-enabled roaches could make a life-saving difference in disaster situations.

Convergence of IT and OT

Until recently, information technology (IT) and operational technology (OT) have for the most part lived in separate worlds. IT supports connections to the Internet along with related data and technology systems and is focused on the secure flow of data across an organization. OT monitors and controls devices and processes on physical operational systems. These systems include assembly lines, utility distribution networks, production facilities, roadway systems, and many more. Typically, IT did not get involved with the production and logistics of OT environments.

Specifically, the IT organization is responsible for the information systems of a business, such as email, file and print services, databases, and so on. In comparison, OT is responsible for the devices and processes acting on industrial equipment, such as factory machines, meters, actuators, electrical distribution automation devices, SCADA (super- visory control and data acquisition) systems, and so on. Traditionally, OT has used dedi- cated networks with specialized communications protocols to connect these devices, and these networks have run completely separately from the IT networks.

Management of OT is tied to the lifeblood of a company. For example, if the network connecting the machines in a factory fails, the machines cannot function, and produc- tion may come to a standstill, negatively impacting business on the order of millions of dollars. On the other hand, if the email server (run by the IT department) fails for a few hours, it may irritate people, but it is unlikely to impact business at anywhere near the same level. Table highlights some of the differences between IT and OT networks and their various challenges.

Comparing Operational Technology (OT) and Information Technology (IT)

Criterion	**Industrial OT Network**	**Enterprise IT Network**
Operational focus	Keep the business operating 24x7	Manage the computers, data, and employee communication system in a secure way
Priorities	1. Availability 2. Integrity	1. Security 2. Integrity

	3. Security	3. Availability
Types of data	Monitoring, control, and supervisory data	Voice, video, transactional, and bulk data
Security	Controlled physical access to devices	Devices and users authenticated to the network
Implication of failure	OT network disruption directly impacts business	Can be business impacting, depending on industry, but workarounds may be Possible

Criterion	Industrial OT Network	Enterprise IT Network
Network upgrades (software or hardware)	Only during operational maintenance windows	Often requires an outage window when workers are not onsite; impact can be mitigated
Security vulnerability	Low: OT networks are isolated and often use proprietary protocols	High: continual patching of hosts is required, and the network is connected to Internet and requires vigilant protection

With the rise of IoT and standards-based protocols, such as IPv6, the IT and OT worlds are converging or, more accurately, OT is beginning to adopt the network protocols, technology, transport, and methods of the IT organization, and the IT organization is beginning to support the operational requirements used by OT. When IT and OT begin using the same networks, protocols, and processes, there are clear economies of scale. Not only does convergence reduce the amount of capital infrastructure needed but networks become easier to operate, and the flexibility of open standards allows faster growth and adaptability to new technologies.

From table, the convergence of IT and OT to a single consolidated network poses several challenges. There are fundamental cultural and priority differences between these two organizations. IoT is forcing these groups to work together, when in the past they have operated rather autonomously. For example, the

OT organization is baffled when IT schedules a weekend shutdown to update software without regard to production requirements. On the other hand, the IT group does not understand the prevalence of proprietary or specialized systems and solutions deployed by OT.

Take the case of deploying quality of service (QoS) in a network. When the IT team deploys QoS, voice and video traffic are almost universally treated with the highest level of service. However, when the OT system shares the same network, a very strong argu- ment can be made that the real-time OT traffic should be given a higher priority than even voice because any disruption in the OT network could impact the business.

With the merging of OT and IT, improvements are being made to both systems. OT is looking more toward IT technologies with open standards, such as Ethernet and IP. At the same time, IT is becoming more of a business partner with OT by better understanding business outcomes and operational requirements.

The overall benefit of IT and OT working together is a more efficient and profitable business due to reduced downtime, lower costs through economy of scale, reduced inventory, and improved delivery

times. When IT/OT convergence is managed correctly, IoT becomes fully supported by both groups. This provides a “best of both worlds” scenario, where solid industrial control systems reside on an open, integrated, and secure technology foundation.

Challenges of IoT

- **IoT Challenges**

Developing a successful IoT application is still not an easy task due to multiple challenges. These challenges include: mobility, reliability, scalability, management, availability, interoperability, and security and privacy. In the following, we briefly describe each of these challenges.

Mobility

IoT devices need to move freely and change their IP address and networks based on their location. Thus, the routing protocol, such as RPL has to reconstruct the DODAG each time a node goes off the network or joins the network which adds a lot of overhead. In addition, mobility might result in a change of service provider which can add another layer of complexity due to service interruption and changing gateway.

Reliability

System should be perfectly working and delivering all of its specifications correctly. It is a very critical requirement in applications that requires emergency responses. In IoT applications, the system should be highly reliable and fast in collecting data, communicating them and making decisions and eventually wrong decisions can lead to disastrous scenarios.

Scalability

Scalability is another challenge of IoT applications where millions and trillions of devices could be connected on the same network. Managing their distribution is not an easy task. In addition, IoT applications should be tolerant of new services and devices constantly joining the network and, therefore, must be designed to enable extensible services and operations.

Management

Managing all These devices and keeping track of the failures, configurations, and performance of such large number of devices is definitely a challenge in IoT. Providers should manage Fault, Configuration, Accounting, Performance and Security (FCAPS) of their interconnected devices and account for each aspect.

Availability

Availability of IoT includes software and hardware levels being provided at anytime and anywhere for service subscribers. Software availability means that the service is provided to anyone who is authorized to have it. Hardware availability means that the existing devices are easy to access and are compatible with IoT functionality and protocols. In addition, these protocols should be compact to be able to be embedded within the IoT constrained devices.

Interoperability

Interoperability means that heterogeneous devices and protocols need to be able to inter-work with each other. This is challenging due to the large number of different platforms used in IoT systems. Interoperability should be handled by both the application developers and the device manufacturers in order to deliver the services regardless of the platform or hardware specification used by the customer.

Challenge	Description
Scale	While the scale of IT networks can be large, the scale of OT can be several orders of magnitude larger. For example, one large electrical utility in Asia recently began deploying IPv6-based smart meters on its electrical grid. While this utility company has tens of thousands of employees (which can be considered IP nodes in the network), the number of meters in the service area is tens of millions. This means the scale of the network the utility is managing has increased by more than 1,000-fold! .
Security	With more "things" becoming connected with other "things" and people, security is an increasingly complex issue for IoT. Your threat surface is now greatly expanded, and if a device gets hacked, its connectivity is a major concern. A compromised device can serve as a launching point to attack other devices and systems. IoT security is also pervasive across just about every facet of IoT. For more information on IoT security
Privacy	As sensors become more prolific in our everyday lives, much of the data they gather will be specific to individuals and their activities. This data can range from health information to shopping patterns and transactions at a retail establishment. For businesses, this data has monetary value. Organizations are now discussing who owns this data and how individuals can control whether it is shared and with whom.
Big data and data analytics	IoT and its large number of sensors is going to trigger a deluge of data that must be handled. This data will provide critical information and insights if it can be processed in an efficient manner. The challenge, however, is evaluating massive amounts of data arriving from different sources in various forms and doing so in a timely manner.

Challenge	Description
Interoperability	As with any other nascent technology, various protocols and architectures are jockeying for market share and standardization within IoT. Some of these protocols and architectures are based on proprietary elements, and others are open. Recent IoT standards are helping minimize this problem, but there are often various protocols and implementations available for IoT networks. The prominent protocols and architectures—especially open, standards-based implementations

TEXT / REFERENCE BOOKS

1. Jan Holler, VlasiosTsiatsis, Catherine Mulligan, Stefan Avesand, StamatisKarnouskos, David Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence Press, 2014.
2. Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.
3. Hassan, Qusay F., ed. Internet of Things A to Z: technologies and applications. John Wiley & Sons, 2018.

4. Holler, Jan, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand, and David Boyle. Internet of Things. Academic Press, 2014.
5. Hersent, Olivier, David Boswarthick, and Omar Elloumi. The internet of things: Key applications and protocols. John Wiley & Sons, 2011.
6. Bernd Scholz- -3-642-19156-5 e-ISBN Architecting the Internet of Things, ISBN 978-3-642-19156-5, 978-3-642-19157-2, Springer

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – II – IoT Architecture and its Protocols– SCSA1408

UNIT 2 IoT REFERENCE ARCHITECTURE

IoT Architecture-State of the Art Introduction, State of the art, Reference Model and architecture, IoT reference Model - IoT Reference Architecture- Introduction, Functional View, Information View, Deployment and Operational View, Other Relevant architectural views. Real-World Design Constraints- Introduction, Technical Design constraints-hardware is popular again, Data representation and visualization, Interaction and remote control

IoT Architecture Overview

IoT can be classified into a four or five-layered architecture which gives you a complete overview of how it works in real life. The various components of the architecture include the following:

Four-layered architecture: this includes media/device layer, network layer, service and application support layer, and application layer.

Five-layered architecture: this includes perception layer, network layer, middleware layer, application layer, and business layer.

Functions of Each Layer

Sensor/Perception layer: This layer comprises of wireless devices, sensors, and radio frequency identification (RFID) tags that are used for collecting and transmitting raw data such as the temperature, moisture, etc. which is passed on to the next layer.

Network layer: This layer is largely responsible for routing data to the next layer in the hierarchy with the help of network protocols. It uses wired and wireless technologies for data transmission.

Middleware layer: This layer comprises of databases that store the information passed on by the lower layers where it performs information processing and uses the results to make further decisions.

Service and application support layer: This layer involve business process modeling andexecution as well as IoT service monitoring and resolution.

Application layer: It consists of application user interface and deals with various applicationssuch as home automation, electronic health monitoring, etc.

Business layer: this layer determines the future or further actions required based on the dataprovided by the lower layers.

Building an IoT ArchitectureBUILDING BLOCKS of IoT

Four things form basic building blocks of the IoT system –sensors, processors, gateways, applications. Each of these nodes has to have its own characteristics in order to form a useful IoT system.

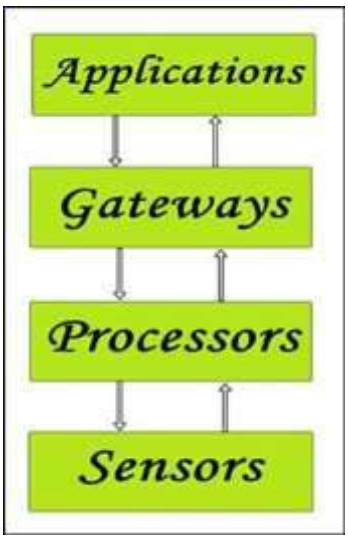


Fig 2.1 Simplified block diagram of the basic building blocks of the IoTSensors:

These form the front end of the IoT devices. These are the so-called "Things" of the system. Their main purpose is to collect data from its surroundings (sensors) or give out data to its surrounding (actuators). These have to be uniquely identifiable devices with a unique IP address so that they can be easily identifiable over a large network. These have to be active in nature which means that they should be able to collect real-time data. These can eitherwork on their own (autonomous in nature) or can be made to work by the user depending on their needs (user-controlled).

Examples of sensors are gas sensor, water quality sensor, moisture sensor, etc.**Processors:**

Processors are the brain of the IoT system. Their main function is to process the data capturedby the sensors and process them so as to extract the valuable data from the enormous amount

of raw data collected. In a word, we can say that it gives intelligence to the data. Processors mostly work on real-time basis and can be easily controlled by applications. These are also responsible for securing the data – that is performing encryption and decryption of data. Embedded hardware devices, microcontroller, etc are the ones that process the data because they have processors attached to it.

Gateways:

Gateways are responsible for routing the processed data and send it to proper locations for its (data) proper utilization. In other words, we can say that gateway helps in to and fro communication of the data. It provides network connectivity to the data. Network connectivity is essential for any IoT system to communicate.

LAN, WAN, PAN, etc are examples of network gateways.**Applications:**

Applications form another end of an IoT system. Applications are essential for proper utilization of all the data collected. These cloud-based applications which are responsible for rendering the effective meaning to the data collected. Applications are controlled by usersand are a delivery point of particular services. Examples of applications are home automation apps, security systems, industrial control hub, etc.

Main design principles of IoT

1. Do your research

When designing IoT-enabled products, designers might make the mistake of forgetting why customers value these products in the first place. That's why it's a good idea to think about the value an IoT offering should deliver at the initial phase of your design. When getting into IoT design, you're not building products anymore. You're building services and experiences that improve people's lives. That's why in-depth qualitative research is the key to figuring out how you can do that. Assume the perspective of your customers to understand what they need and how your IoT implementation can solve their pain points. Research your target audience deeply to see what their existing experiences are and what they wish was different about them.

2. Concentrate on value

Early adopters are eager to try out new technologies. But the rest of your customer base might be reluctant to put a new solution to use. They may not feel confident with it and are likely to be cautious about using it. If you want your IoT solution to become widely adopted, you need to focus on the actual tangible value it's going to deliver to your target audience. What is the real end-user value of your solution? What might be the barriers to adopting new technology? How can your solution address them specifically? Note that the features the early tech adopters might find valuable might turn out to be completely uninteresting for the majority of users. That's why you need to carefully plan which features to include and in what order, always concentrating on the actual value they provide.

3. Don't forget about the bigger picture

One characteristic trait of IoT solutions is that they typically include multiple devices that come with different capabilities and consist of both digital and physical touchpoints. Your solution might also be delivered to users in cooperation with service providers. That's why it's not enough to design a single touchpoint well. Instead, you need to take the bigger picture into account and treat your IoT system holistically. Delineate the role of every device and service. Develop a conceptual model of how users will perceive and understand the system. All the parts of your system need to work seamlessly together. Only then you'll be able to create a meaningful experience for your end-users.

4. Remember about the security

Don't forget that IoT solutions aren't purely digital. They're located in the real-world context, and the consequences of their actions might be serious if something goes wrong. At the same time, building trust in IoT solutions should be one of your main design drivers. Make sure that every interaction with your product builds consumer trust rather than breaking it. In practice, it means that you should understand all the possible error situations that maybe related to the context of its use. Then try to design your product in a way to prevent them. If error situations occur, make sure that the user is informed appropriately and provided with help. Also, consider data security and privacy as a key aspect of your implementation. Users need to feel that their data is safe, and objects located in their workspaces or home can't be hacked. That's why quality assurance and testing the system in the real-world context are so important.

5. Build with the context in mind

And speaking of context, it pays to remember that IoT solutions are located at the intersection of the physical and digital world. The commands you give through digital interfaces produce real-world effects. Unlike digital commands, these actions may not be easily undone. In a real-world context, many unexpected things may happen. That's why you need to make sure that the design of your solution enables users to feel safe and in control at all times. The context itself is a crucial consideration during IoT design. Depending on the physical context of your solution, you might have different goals in mind. For example, you might want to minimize user distraction or design devices that will be resistant to the changing weather conditions. The social context is an important factor, as well. Don't forget that the devices you design for workspaces or homes will be used by multiple users.

6. Make good use of prototypes

IoT solutions are often difficult to upgrade. Once the user places the connected objectsomewhere, it might be hard to replace it with a new version – especially if the user would have to pay for the upgrade. Even the software within the object might be hard to update because of security and privacy reasons. Make sure that your design practices help to avoid costly hardware iterations. Get your solution right from the start. From the design perspective, it means that prototyping and rapid iteration will become critical in the early stages of the project.

Standards consideration for IoT

Alliances have been formed by many domestic and multinational companies to agree on common standards and technology for the IoT. However, no universal body has been formed

yet. While organizations such as IEEE, Internet Engineering Task Force (IETF), ITU-T, OneM2M, 3GPP, etc., are active at international level, Telecommunication Standards Development Society, India (TSDSI), Global ICT Standardization Forum for India (GISFI), Bureau of Indian Standards (BIS), Korean Agency for Technology and Standards (KATS), and so on, are active at national level and European Telecommunications Standards Institute (ETSI) in the regional level for standardization.

IoT Architecture

Introduction

The Internet of Things (IoT) has seen an increasing interest in adaptive frameworks and architectural designs to promote the correlation between IoT devices and IoT systems. This is because IoT systems are designed to be categorized across diverse application domains and geographical locations. It, therefore, creates extensive dependencies acrossdomains, platforms and services. Considering this interdependency between IoT devices and IoT systems, an intelligent, connection-aware framework has become a necessity, this is where IoT architecture comes into play! Imagine a variety of smart IoT systems from sensors and actuators to internet getaways and Data Acquisition Systems all under the centralized control of one “brain”! The brain here can be referred to as the IoT architecture, whose effectiveness and applicability directly correlate with the quality of its building blocks. The way a system interacts and the different functions an IoT device performs are various approaches to IoT architecture. Since we can call the architecture the brain, it’s also possible to say that the key causes of poor integration in IoT systems are the shortage of intelligent, connection-aware architecture to support interaction in IoT systems.

An IoT architecture is the system of numerous elements that range from sensors, protocols, actuators, to cloud services, and layers. Besides, devices and sensors the Internet of Things (IoT) architecture layers are distinguished to track the consistency of a system through protocols and gateways. Different architectures have been proposed by researchers and we can all agree that there is no single consensus on architecture for IoT. The most basic architecture is a three-layer architecture.

State-of-the-art

The IoT can be considered both a dynamic and global networked infrastructure that manages self-configuring objects in a highly intelligent way. This, in turn, allows the interconnection of IoT devices that share their information to create new applications and services which can improve human lives. Originally, the concept of the IoT was first introduced by Kevin Ashton, who is the founder of MIT auto-identification centre

in 1999. Ashton has said, “The Internet of Things has the potential to change the world, just as the Internet did. Maybe even more so”. Later, the IoT was officially presented by the International Telecommunication Union (ITU) in 2005. The IoT has many definitions suggested by many organizations and researchers. However, the definition provided by ITU in 2012 is the most common. It stated: “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies”. In addition, Guillemin and Friess in have suggested one of the simplest definitions that describe the IoT in a smooth manner. It stated: “The Internet of Things allows people and things to be connected Anytime, Anyplace, with anything and anyone, ideally using any path/network and any service”. Several definitions were suggested by many researchers describing the IoT system from different perspectives but the important thing that majority or researchers have agreed on is the IoT is created for a better world for all the human beings. The IoT is a promising technology that starts to grow significantly. There were already more objects/things connected to the Internet than people from 2008. Predictions are made that in the future; the number of Internet-connected devices will reach or even exceed 50 billion. In addition, the IoT becomes the most massive device market that enables companies to save billions of dollars. It has added $1.7 trillion in value to the global economy in 2019. This involves hardware, software, management services, installation costs, and economic value from realized IoT efficiencies. Nowadays, the IoT notion has evolved to include the perception of realizing a global infrastructure of interconnected networks of physical and virtual objects. The huge technological development has expanded the idea of the IoT to involve other technologies such as Cloud computing and Wireless Sensor Networks (WSNs). The IoT has become able to connect both humans and things anywhere, and anytime, ideally using any path/network. The IoT has become one of the interesting topics to many researchers. According to Google, the number of IoT journal and conference papers has almost doubled from 2004 to 2010. From 2010, the IoT articles are dramatically increased to reach about 985 articles in 2015.

Architecture Reference ModelIntroduction

A reference model is a division of functionality together with data flow between the pieces. A reference model is a standard decomposition of a known problem into parts that cooperatively solve the problem. Arising from experience, reference models are a characteristic of mature domains. Can you name the standard parts of a compiler or adatabase management system? Can you explain in broad terms how the parts work together toaccomplish their collective purpose? If so, it is because you have been taught a reference model of these applications.

A reference architecture is a reference model mapped onto software elements (that cooperatively implement the functionality defined in the reference model) and the data flows between them. Whereas a reference model divides the functionality, a reference architecture is the mapping of that functionality onto a system decomposition. The mapping may be, but by no means necessarily is, one to one. A software element may implement part of a function or several functions. Reference models, architectural patterns, and reference architectures arenot architectures; they are useful concepts that capture elements of an architecture. Each is the outcome of early design decisions. The relationship among these design elements is shown in Figure 1

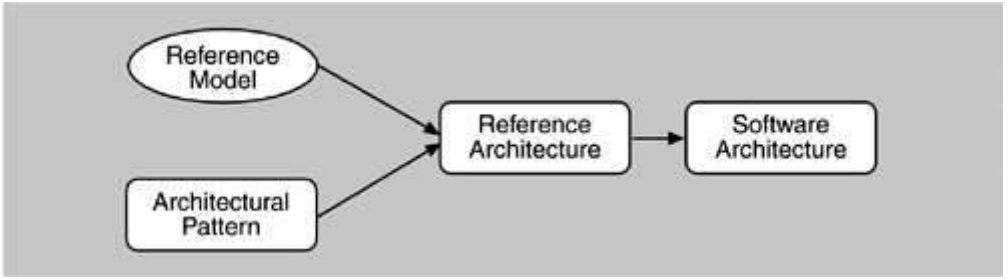


Fig 2.2 The relationships of reference models, architectural patterns, reference architectures,and software architectures.

IoT Reference Architecture

The reference architecture consists of a set of components. Layers can be realized by means of specific technologies, and we will discuss options for realizing each component. There are also some cross-cutting/vertical layers such as access/identity management.

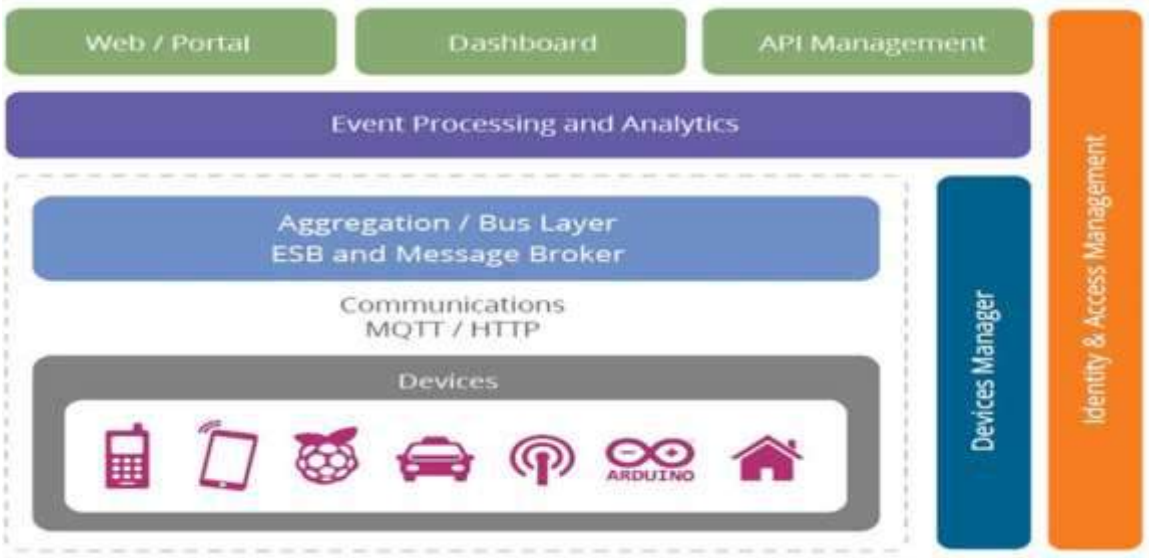


Fig 2.3 IoT Reference Architecture

The layers are

- Client/external communications - Web/Portal, Dashboard, APIs
- Event processing and analytics (including data storage)
- Aggregation/bus layer – ESB and message broker
- Relevant transports - MQTT/HTTP/XMPP/CoAP/AMQP, etc.
- Devices

The cross-cutting layers are

- Device manager
- Identity and access managements

THE DEVICE LAYER

The bottom layer of the architecture is the device layer. Devices can be of various types, but in order to be

considered as IoT devices, they must have some communications that either indirectly or directly attaches to the Internet. Examples of direct connections are

- Arduino with Arduino Ethernet connection
- Arduino Yun with a Wi-Fi connection
- Raspberry Pi connected via Ethernet or Wi-Fi
- Intel Galileo connected via Ethernet or

Wi-Fi Examples of indirectly connected devices include

- ZigBee devices connected via a ZigBee gateway
- Bluetooth or Bluetooth Low Energy devices connecting via a mobile phone
- Devices communicating via low power radios to a

Raspberry Pi There are many more such examples of each type.

Each device typically needs an identity. The identity may be one of the following:

- A unique identifier (UUID) burnt into the device (typically part of the System-on Chip, orprovided by a secondary chip) •

A UUID provided by the radio subsystem (e.g. Bluetooth identifier, Wi-Fi MAC address)

- An OAuth2 Refresh/Bearer Token (this may be in addition to one of the above)
- An identifier stored in nonvolatile memory such as EEPROM

For the reference architecture we recommend that every device has a UUID (preferably an unchangeable ID provided by the core hardware) as well as an OAuth2 Refresh and Bearer token stored in EEPROM. The specification is based on HTTP; however, (as we will discuss in the communications section) the reference architecture also supports these flows over MQTT.

COMMUNICATIONS LAYER

The communication layer supports the connectivity of the devices. There are multiple potential protocols for communication between the devices and the cloud.

The most well-known three potential protocols are

- HTTP/HTTPS (and RESTful approaches on those)
- MQTT 3.1/3.1.1(Message Queuing Telemetry Transport)
- Constrained application protocol (CoAP)

HTTP is well known, and there are many libraries that support it. Because it is a simple text- based protocol, many small devices such as 8-bit controllers can only partially support the protocol – for example enough code to POST or GET a resource. The larger 32-bit based devices can utilize full HTTP client libraries that properly implement the whole protocol. There are several protocols optimized for IoT use. The two best known are MQTT6 and CoAP7. MQTT was invented in 1999 to solve issues in embedded systems and SCADA. It has been through some iterations and the current version (3.1.1) is undergoing standardizationin the OASIS MQTT Technical Committee8. MQTT is a publish-subscribe messaging systembased on a broker model. The protocol

has a very small overhead (as little as 2 bytes per message), and was designed to support lossy and intermittently connected networks. MQTT was designed to flow over TCP. In addition, there is an associated specification designed for ZigBee-style networks called MQTT-SN (Sensor Networks). CoAP is a protocol from the IETF that is designed to provide a RESTful application protocol modeled on HTTP semantics, but with a much smaller footprint and a binary rather than a text- based approach. CoAP is a more traditional client-server approach rather than a brokered approach. CoAP is designed to be used over UDP. For the reference architecture we have opted to select MQTT as the preferred device communication protocol, with HTTP as an alternative option.

The reasons to select MQTT and not CoAP at this stage are

- Better adoption and wider library support for MQTT;
- Simplified bridging into existing event collection and event processing systems; and
- Simpler connectivity over firewalls and NAT networks

However, both protocols have specific strengths (and weaknesses) and so there will be some situations where CoAP may be preferable and could be swapped in. In order to support MQTT we need to have an MQTT broker in the architecture as well as device libraries. We will discuss this with regard to security and scalability later. One important aspect with IoT devices is not just for the device to send data to the cloud/ server, but also the reverse. This is one of the benefits of the MQTT specification: because it is a brokered model, clients connectan outbound connection to the broker, whether or not the device is acting as a publisher or subscriber. This usually avoids firewall problems because this approach works even behind firewalls or via NAT. In the case where the main communication is based on HTTP, the traditional approach for sending data to the device would be to use HTTP Polling. This isvery inefficient and costly, both in terms of network traffic as well as power requirements. The modern replacement for this is the WebSocket protocol9 that allows an HTTP connection to be upgraded into a full two-way connection. This then acts as a socket channel (similar to a pure TCP channel) between the server and client. Once that has been established, it is up to the system to choose an ongoing protocol to tunnel over the connection. For the reference architecture we once again recommend using MQTT as a protocol with WebSockets. In some cases, MQTT over Web Sockets will be the only protocol. This is because itis even more firewall-friendly than the base MQTT specification as well as supporting pure browser/JavaScript clients using the same protocol. Note that while there is some support for Web Sockets on small controllers, such as Arduino, the combination of network code, HTTP and Web Sockets would utilize most of the available code space on a typical Arduino 8-bit device. Therefore, it is recommended the use of Web Sockets on the larger 32-bit devices.

AGGREGATION/BUS LAYER

An important layer of the architecture is the layer that aggregates and brokers communications. This is an important layer for three reasons:

1. The ability to support an HTTP server and/or an MQTT broker to talk to the devices

2. The ability to aggregate and combine communications from different devices and to route communications to a specific device (possibly via a gateway)

3. The ability to bridge and transform between different protocols, e.g. to offer HTTP based APIs that are mediated into an MQTT message going to the device. The aggregation/bus layer provides these capabilities as well as adapting into legacy protocols. The bus layer may also provide some simple correlation and mapping from different correlation models (e.g. mapping a device ID into an owner‘s ID or vice-versa). Finally, the aggregation/bus layer needs to perform two key security roles. It must be able to act as an OAuth2 Resource

Server (validating Bearer Tokens and associated resource access scopes). It must also be able to act as a policy enforcement point (PEP) for policy-based access. In this model, the bus makes requests to the identity and access management layer to validate access requests. The identity and access management layer acts as a policy decision point (PDP) in this process. The bus layer then implements the results of these calls to the PDP to either allow or disallowresource access.

EVENT PROCESSING AND ANALYTICS LAYER

This layer takes the events from the bus and provides the ability to process and act upon these events. A core capability here is the requirement to store the data into a database. This may happen in three forms. The traditional model here would be to write a server-side application,
e.g. this could be a JAX-RS application backed by a database. However, there are many approaches where we can support more agile approaches. The first of these is to use a big data analytics platform. This is a cloudscalable platform that supports technologies such as Apache Hadoop to provide highly scalable map reduce analytics on the data coming from the devices. The second approach is to support complex event processing to initiate near real- time activities and actions based on data from the devices and from the rest of the system.

Our recommended approach in this space is to use the following approaches:

- Highly scalable, column-based data storage for storing events
- Map-reduce for long-running batch-oriented processing of data
- Complex event processing for fast in-memory processing and near real-time reaction and autonomic actions based on the data and activity of devices and other systems
- In addition, this layer may support traditional application processing platforms, such as Java Beans, JAX-RS logic, message-driven beans, or alternatives, such as node.js, PHP, Ruby or Python.

CLIENT/EXTERNAL COMMUNICATIONS LAYER

The reference architecture needs to provide a way for these devices to communicate outside of the device-oriented system. This includes three main approaches. Firstly, we need the ability to create web-based front-ends and portals that interact with devices and with the event-processing layer. Secondly, we need the ability to create dashboards that offer views into analytics and event processing. Finally, we need to be able to interact with systems outside this network using machine-to-machine communications (APIs). These APIs need to be managed and controlled and this happens in an API management system. The recommended approach to building the web front end is to utilize a modular front-end architecture, such as a portal, which allows simple fast composition of useful UIs. Of course, the architecture also supports existing Web server-side technology, such as Java Servlets/ JSP, PHP, Python, Ruby, etc. Our recommended approach is based on the Java framework and the most popular Java-based web server, Apache Tomcat. The dashboard is a re-usable system focused on creating graphs and other visualizations of data coming from the devices and the event processing layer.

The API management layer provides three main functions:

- The first is that it provides a developer-focused portal (as opposed to the user focused portal previously mentioned), where developers can find, explore, and subscribe to APIs from the system. There is also support for publishers to create, version, and manage the available and published APIs;
- The second is a gateway that manages access to the APIs, performing access control checks (for external requests) as well as throttling usage based on policies. It also performs routing and load- balancing;

- The final aspect is that the gateway publishes data into the analytics layer where it is stored as well as processed to provide insights into how the APIs are used.

DEVICE MANAGEMENT

Device management (DM) is handled by two components. A server-side system (the device manager) communicates with devices via various protocols and provides both individual and bulk control of devices. It also remotely manages software and applications deployed on the device. It can lock and/or wipe the device if necessary. The device manager works in conjunction with the device management agents. There are multiple different agents for different platforms and device types. The device manager also needs to maintain the list of device identities and map these into owners. It must also work with the identity and access management layer to manage access controls over devices (e.g. who else can manage the device apart from the owner, how much control does the owner have vs. the administrator, etc.) There are three levels of device: non-managed, semi-managed and fully managed (NM, SM, FM). Fully managed devices are those that run a full DM agent.

A full DM agent supports:

- Managing the software on the device
- Enabling/disabling features of the device (e.g. camera, hardware, etc.)
- Management of security controls and identifiers
- Monitoring the availability of the device • Maintaining a record of the device location ifavailable
- Locking or wiping the device remotely if the device is compromised, etc.

Non-managed devices can communicate with the rest of the network, but have no agent involved. These may include 8-bit devices where the constraints are too small to support the agent. The device manager may still maintain information on the availability and location of the device if this is available. Semi-managed devices are those that implement some parts of the DM (e.g. feature control, but not software management).

IDENTITY AND ACCESS MANAGEMENT

The final layer is the identity and access management layer. This layer needs to provide the following services:

- OAuth2 token issuing and validation
- Other identity services including SAML2 SSO and OpenID Connect support for identifyinginbound requests from the Web layer
- XACML PDP
- Directory of users (e.g. LDAP)
- Policy management for access control (policy control point)

The identity layer may of course have other requirements specific to the other identity and access management for a given instantiation of the reference architecture. In this section we have outlined the major components of the reference architecture as well as specific decisions we have taken around technologies. These decisions are motivated by the specific requirements of IoT architectures as well as best practices for building agile, evolvable, scalable Internet architectures.

IoT Reference Model

In an IoT system, data is generated by multiple kinds of devices, processed in different ways, transmitted to different locations, and acted upon by applications. The proposed IoT reference model is comprised of seven levels. Each level is defined with terminology that can be standardized to create a globally accepted frame of reference. The IoT Reference Model does not restrict the scope or locality of its components. For example, from a physical perspective, every element could reside in a single rack of equipment or it could be distributed across the world. The IoT Reference Model also allows the processing occurring at each level to range from trivial to complex, depending on the situation. The model describes how tasks at each level should be handled to maintain simplicity, allow high scalability, and ensure supportability. Finally, the model defines the functions required for an IoT system to be complete. Figure illustrates the IoT Reference model and its levels. It is important to note that in the IoT, data flows in both directions. In a control pattern, control information flows from the top of the model (level 7) to the bottom (level 1). In a monitoring pattern, the flowof information is the reverse. In most systems, the flow will be bidirectional.

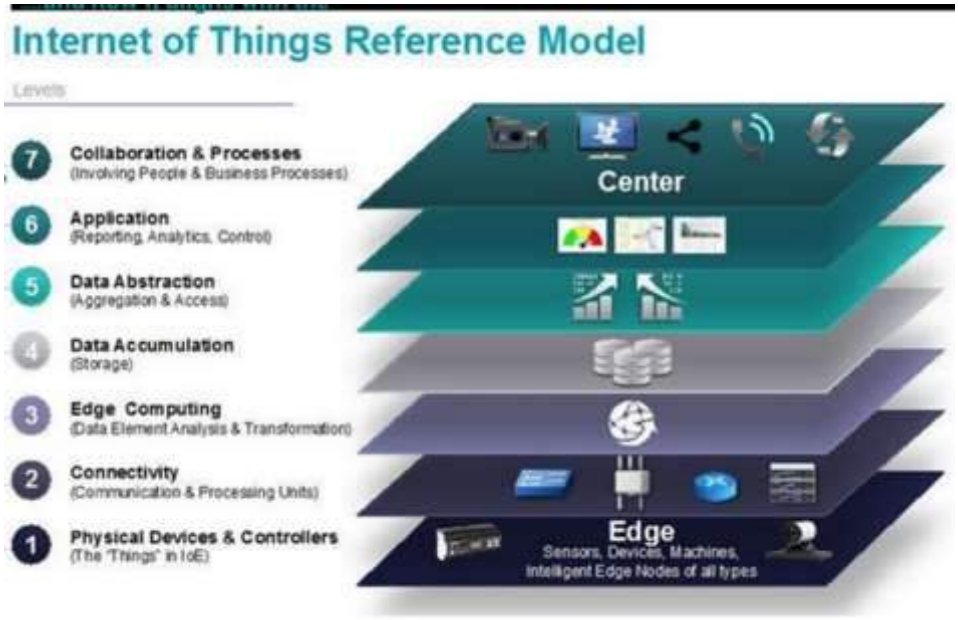


Fig 2.4 IoT Reference Model

Level 1: Physical Devices and Controllers

The IoT Reference Model starts with Level 1: physical devices and controllers that might control multiple devices. These are the "things" in the IoT, and they include a wide range of endpoint devices that send and receive information. Today, the list of devices is already extensive. It will become almost unlimited as more equipment is added to the IoT over time. Devices are diverse, and there are no rules about size, location, form factor, or origin. Some devices will be the size of a silicon chip. Some will be as large as vehicles. The IoT must support the entire range. Dozens or hundreds of equipment manufacturers will produce IoT devices. To simplify compatibility and support manufacturability, the IoT Reference Model generally describes the level of processing needed from Level 1 devices.

Level 2: Connectivity

Communications and connectivity are concentrated in one level—Level 2.

The most important function of Level 2 is reliable, timely information transmission. This includes transmissions: ● Between devices (Level 1) and the network ● Across networks (east-west) ● Between the network (Level 2) and low-level information processing occurring at Level 3

Traditional data communication networks have multiple functions, as evidenced by the International Organization for Standardization (ISO) 7-layer reference model. However, a complete IoT system contains many levels in addition to the communications network. One objective of the IoT Reference Model is for communications and processing to be executed by existing networks. The IoT Reference Model does not require or indicate creation of a different network—it relies on existing networks. However, some legacy devices aren't IP- enabled, which will require introducing communication gateways. Other devices will require proprietary controllers to serve the communication function. However, over time, standardization will increase. As Level 1 devices proliferate, the ways in which they interact with Level 2 connectivity equipment may change. Regardless of the details, Level 1 devices communicate through the IoT system by interacting with Level 2 connectivity equipment

Level 3: Edge (Fog) Computing

The functions of Level 3 are driven by the need to convert network data flows into information that is suitable for storage and higher-level processing at Level 4 (data accumulation). This means that Level 3 activities focus on high-volume data analysis and transformation. For example, a Level 1 sensor device might generate data samples multiple times per second, 24 hours a day, 365 days a year. A basic tenet of the IoT Reference Model is that the most intelligent system initiates information processing as early and as close to the edge of the network as possible. This is sometimes referred to as fog computing. Level 3 is where this occurs. Given that data is usually submitted to the connectivity level (Level 2) networking equipment by devices in small units, Level 3 processing is performed on a packet-by-packet basis. This processing is limited, because there is only awareness of data units—not "sessions" or "transactions." Level 3 processing can encompass many examples, such as:

- Evaluation: Evaluating data for criteria as to whether it should be processed at a higherlevel
- Formatting: Reformatting data for consistent higher-level processing
- Expanding/decoding: Handling cryptic data with additional context (such as the origin)
- Distillation/reduction: Reducing and/or summarizing data to minimize the impact of data and traffic on the network and higher-level processing systems
- Assessment: Determining whether data represents a threshold or alert; this could include redirecting data to additional destinations

Level 4: Data Accumulation

Networking systems are built to reliably move data. The data is "in motion." Prior to Level 4, data is moving through the network at the rate and organization determined by the devices generating the data. The model is event driven. As defined earlier, Level 1 devices do not include computing capabilities themselves. However, some computational activities could occur at Level 2, such as protocol translation or application of network security policy. Additional compute tasks can be performed at Level 3, such as packet inspection. Driving computational tasks as close to the edge of the IoT as possible, with heterogeneous systems distributed across multiple management domains represents an example of fog computing.

Fog computing and fog services will be a distinguishing characteristic of the IoT. Most applications cannot, or do not need to, process data at network wire speed. Applications typically assume that data is "at rest"—or unchanging—in memory or on disk. At Level 4, Data Accumulation, data in motion is converted to data at rest.

Level 4 determines:

- If data is of interest to higher levels: If so, Level 4 processing is the first level that is configured to serve the specific needs of a higher level.
- If data must be persisted: Should data be kept on disk in a non-volatile state or accumulated in memory for short-term use?
- The type of storage needed: Does persistency require a file system, big data system, or relational database?
- If data is organized properly: Is the data appropriately organized for the required storage system?
- If data must be recombined or recomputed: Data might be combined, recomputed, or aggregated with previously stored information, some of which may have come from non-IoT sources.

As Level 4 captures data and puts it at rest, it is now usable by applications on a non-real- time basis. Applications access the data when necessary. In short, Level 4 converts event- based data to query-based processing. This is a crucial step in bridging the differences between the real-time networking world and the non-real-time application world.

Level 5: Data Abstraction

IoT systems will need to scale to a corporate—or even global—level and will require multiple storage systems to accommodate IoT device data and data from traditional enterprise ERP, HRMS, CRM, and other systems. The data abstraction functions of Level 5 are focused on rendering data and its storage in ways that enable developing simpler, performance- enhanced applications.

With multiple devices generating data, there are many reasons why this data may not land in the same data storage:

- There might be too much data to put in one place.
- Moving data into a database might consume too much processing power, so that retrieving it must be separated from the data generation process. This is done today with online transaction processing (OLTP) databases and data warehouses.
- Devices might be geographically separated, and processing is optimized locally.
- Levels 3 and 4 might separate "continuous streams of raw data" from "data that represents an event." Data storage for streaming data may be a big data system, such as Hadoop. Storage for event data may be a relational database management system (RDBMS) with faster query times.
- Different kinds of data processing might be required.

For example, in-store processing will focus on different things than across-all-stores summary processing. For these reasons, the data abstraction level must process many different things. These include:

- Reconciling multiple data formats from different sources
- Assuring consistent semantics of data across sources
- Confirming that data is complete to the higher-level application

- Consolidating data into one place (with ETL, ELT, or data replication) or providing accessto multiple data stores through data virtualization
- Protecting data with appropriate authentication and authorization
- Normalizing or denormalizing and indexing data to provide fast application access Level

Application Level 6

It is the application level, where information interpretation occurs. Software at this level interacts with Level 5 and data at rest, so it does not have to operate at network speeds. The IoT Reference Model does not strictly define an application. Applications vary based on vertical markets, the nature of device data, and business needs. For example, some applications will focus on monitoring device data. Some will focus on controlling devices. Some will combine device and non-device data. Monitoring and control applications represent many different application models, programming patterns, and software stacks, leading to discussions of operating systems, mobility, application servers, hypervisors, multi- threading, multi-tenancy, etc. These topics are beyond the scope of the IoT Reference Model discussion. Suffice it to say that application complexity will vary widely.

Examples include:

- Mission-critical business applications, such as generalized ERP or specialized industrysolutions
- Mobile applications that handle simple interactions
- Business intelligence reports, where the application is the BI server
- Analytic applications that interpret data for business decisions
- System management/control center applications that control the IoT system itself and don'tact on the data produced by it

If Levels 1-5 are architected properly, the amount of work required by Level 6 will be reduced. If Level 6 is designed properly, users will be able to do their jobs better.

Level 7: Collaboration and Processes

One of the main distinctions between the Internet of Things (IoT) and IoT is that IoT includes people and processes. This difference becomes particularly clear at Level 7: Collaboration and Processes. The IoT system, and the information it creates, is of little value unless it yieldsaction, which often requires people and processes. Applications execute business logic to empower people. People use applications and associated data for their specific needs. Often, multiple people use the same application for a range of different purposes. So, the objective isnot the application—it is to empower people to do their work better. Applications (Level 6) give business people the right data, at the right time, so they can do the right thing. But frequently, the action needed requires more than one person. People must be able to communicate and collaborate, sometimes using the traditional Internet, to make the IoT useful. Communication and collaboration often require multiple steps. And it usually transcends multiple applications. This is why Level 7, represents a higher level than a single application.

IoT Reference Architecture

- Reference Architecture is a starting point for generating concrete architectures and actual systems. A concrete

architecture addresses the concerns of multiple stakeholders of the actual system, and it is typically presented as a series of views thataddress different stakeholder concerns.

- A Reference Architecture, on the other hand, serves as a guide for one or more concrete system architects. However, the concept of views for the presentation of an architecture is also useful for the IoT Reference Architecture.
- Views are useful for reducing the complexity of the Reference Architecture blueprintsby addressing groups of concerns one group at a time.
- However, since the IoT Reference Architecture does not contain details about the environment where the actual system is deployed, some views cannot be presented in detail or at all; for example, the view that shows the concrete Physical Entities and Devices for a specific scenario.
- The stakeholders for a concrete IoT system are the people who use the system(Human Users); the people who design, build, and test the Resources , Services, Active Digital Artifacts, and Applications; the people who deploy Devices and attach them to Physical Entities; the people who integrate IoT capabilities of functions with an existing ICT system (e.g. of an enterprise); the people who operate, maintain, and troubleshoot the Physical and Virtual Infrastructure; and the people who buy and own an IoT system or parts thereof (e.g. city authorities).
- In order to address the concerns of mainly the concrete IoT architect, and secondly theconcerns of most of the above stakeholders, we have chosen to present the Reference Architecture as a set of architectural views .
- Functional View: Description of what the system does, and its main functions.
- Information View: Description of the data and information that the system handles.
- Deployment and Operational View: Description of the main real world components ofthe system such as devices, network routers, servers, etc.

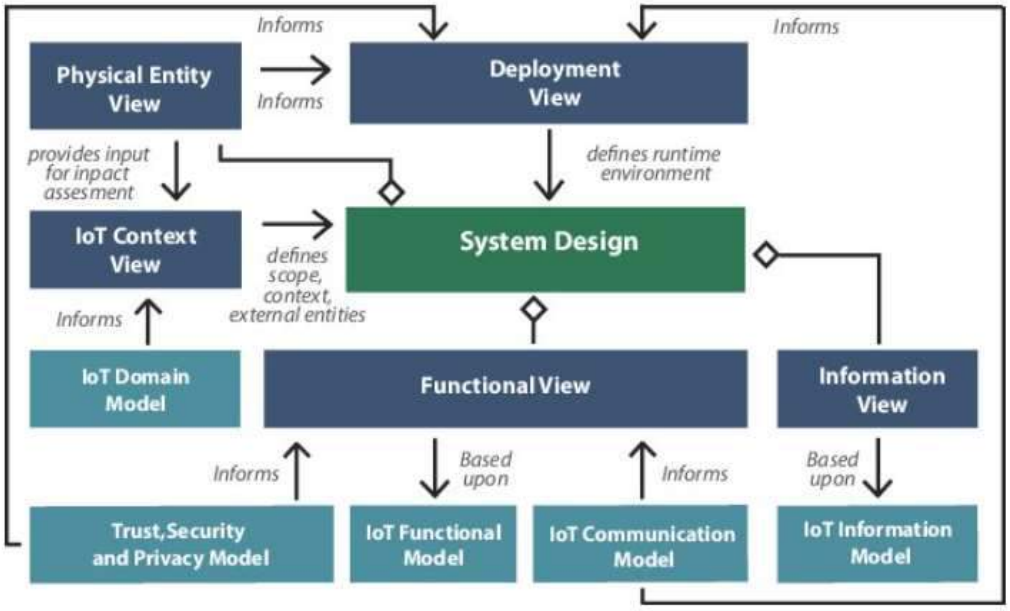


Fig: 2.5 Various views in IoT

● The **Development View** defines how to implement the system.

The Development View addresses the concerns of developers and testers.

○ All software projects involve some amount of new code being written.

○ This view provides a stable environment for more detailed design work.

● The Deployment View defines how to transition the system to live operation.

Focuses on aspects of the system important after the system has been built and is ready to be put into live operation.

● Defines: ○ The physical environment it will run in. ■ Hardware and hosting environment (processing nodes, network interconnections, disk storage). ○ Technical environment requirements for each processing node. ○ Mapping of elements to the runtime environment that will execute them.

It is needed when the system has…

○ Complex runtime dependencies. ■ Third party libraries, network services. ○ Complex runtime environments. ■ Elements distributed across many machines. ○ Dependencies on unfamiliar HW/SW. ■ Deployed on cloud hardware. ● When the system will be deployed in… ○ Wildly varying software environments. ■ Commercial software run on a PC. ○ Wildly varying physical environments. ■ Specialist or unfamiliar hardware.

● **The Operational View** defines how to keep the system alive in the field

Identifies a system-wide strategy for addressing operational concerns. ○ Helps to ensure system is a reliable and effective part of its environment. ○ For packaged software, helps illustrate the types of issues that could occur once installed. ○ Documents how the system can be architected to reduce or address these concerns. ● Often least well-defined view, as many of the details are not fully-defined until construction is underway

Installation and upgrade

○ Team performs the install. ○ Users install and configure themselves. ○ Resources allocated to a cloud environment. ● Is this a pure installation or an upgrade? ○ Upgrades can be more complex. ○ Must respect existing data and settings, state of running elements. ○ Can you keep the system running during update? ● Ensure the system can be installed or updated successfully.

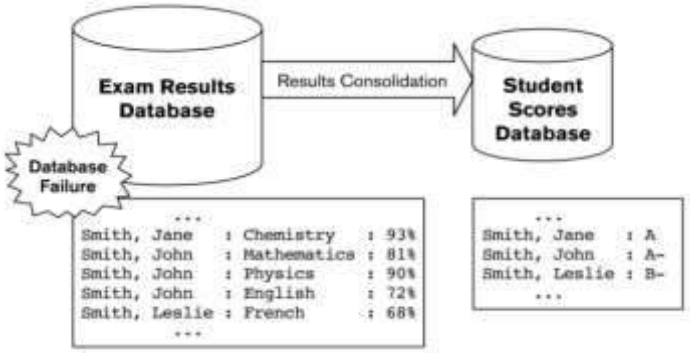
Documenting Installation and Upgrade ● Help the reader understand: ○ What needs to be installed or upgraded to move the system into production. ○ What dependencies exist between groups of items to be installed or upgraded (determines event order). ○ What constraints exist on the installation process. ○ What needs to be done to abandon and undo the installation/upgrade if there is a problem. ● Do not need a complete guide. ○ Instead, constraints the architecture imposes on installation and upgrade.

Operational Monitoring and Control ● Systems require routine monitoring. ● Control operations can be used to keep the system running correctly. ○ Startup, shutdown, transaction resubmission. ● How much is required depends on how many unexpected operational conditions are likely to occur. ● Balance against cost and time. ● Consider deployment environment to identify solutions.

Alerting ● A system should send notifications when something bad happens. ○ Technical: Unable to connect to database. ○ Functional: Bad data on an automated input. ○ Significant non-error conditions (startup, shutdown) ● Active function of a system. ○ Sent to appropriate humans for action. ● Define which events require alerts, what information should be included, and where it should be sent. ● Avoid sending too many alerts

Backup and Restore ● Data must be protected and insured. ○ Backup processes should be designed, built, and tested regularly. ● It must be possible to restore data from a backup in a transactionally consistent state. ○ All

updates committed to the restored database or not recovered at all. ○ Consider data lost as part of restoring (at least any transactions active during failure). ● Failure in one element could corrupt system. ○ Recover or recreate lost data. ○ Revert system to older state.



Academic records in databases. ○ Exam results database. ○ Scores database transforms data into a overall score. ● Corruption requires restoration of exam database. ○ Over three months old. ○ Results from those months will need to be reentered. ○ However, student scores already reflect that data. Must prevent reentered data from changing scores.

Documenting System Administration ● Monitoring and control facilities ○ How to monitor and adjust the system. ○ Custom utilities, existing management environments. ○ Basic message log to full-blown infrastructure. ○ Define what features you will offer, how to use them, and any limitations. ● Required routine procedures ○ What needs to be performed regularly? ○ Backup and health check procedures. ○ Define purpose of each procedure, when performed, who performs it, and the steps involved.

Likely error conditions ○ Error conditions may require administrative intervention (disk full, network failure). ○ What is unique about architecture? ○ Explain error conditions, when they occur, how to recognize them, and HOW to correct them. ● Performance monitoring facilities ○ Watch the system for performance problems. ○ Extracted and analyzed routinely. ○ Explain measures taken, how they can be extracted and analyzed.

Functional view

- The functional view for the IoT Reference Architecture is presented in Figure.5.1 ,and is adapted from IoT-A .
- It consists of the Functional Groups (FGs) presented earlier in the IoT FunctionalModel, each of which includes a set of Functional Components (FCs).
- It is important to note that not all the FCs are used in a concrete IoT architecture, andtherefore the actual system as explained earlier

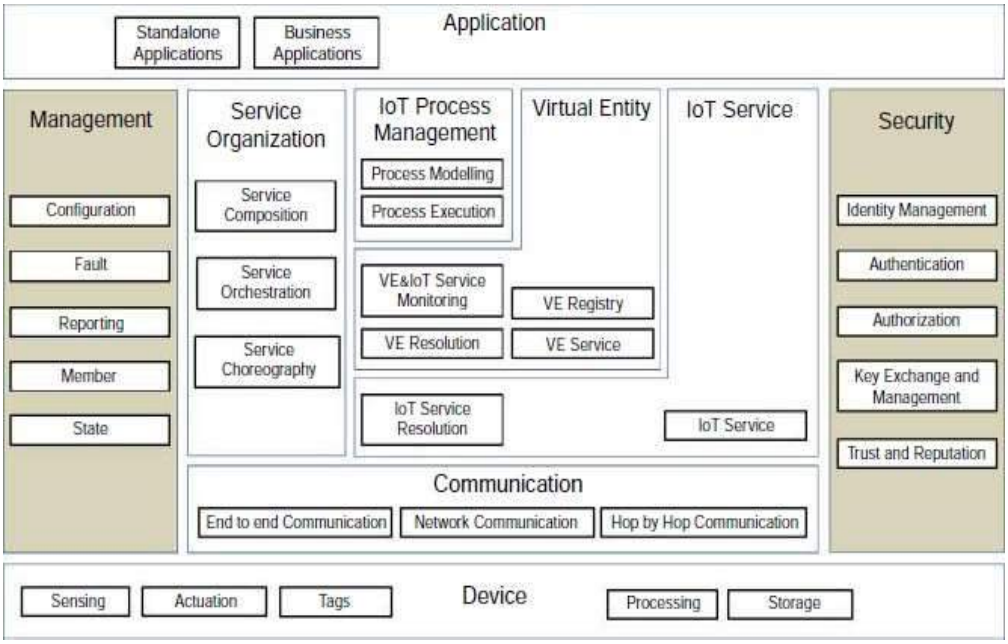


Fig.2.6 IoT Functional View

Device and Application functional group

- The Device and Application FGs are already covered in the IoT Functional Model. For convenience the Device FG contains the Sensing, Actuation, Tag, Processing, Storage FCs, or simply components.
- These components represent the resources of the device attached to the Physical Entities of interest. The Application FG contains either standalone applications (e.g. for iOS, Android, Windows phone), or Business Applications that connect the IoT system to an Enterprise system.

Communication functional group

The Communication FG contains the End-to-End Communication, Network Communication, and Hop-by-Hop communication components:

- The Hop-by-Hop Communication is applicable in the case that devices are equipped with mesh radio networking technologies such as IEEE 802.15.4 for which messages have to traverse the mesh from node-to-node (hop-by-hop) until they reach a gateway node which forwards the message (if needed) further to the Internet.
- The hop-by-hop FC is responsible for transmission and reception of physical and MAC layer frames to/from other devices. This FC has two main interfaces: (a) one "southbound" to/from the actual radio on the device, and (b) one "northbound" to/from the Network FC in the Communication FG.
- The Network FC is responsible for message routing & forwarding and the necessary translations of various identifiers and addresses.
- The translations can be (a) between network layer identifiers to MAC and/or physical network identifiers, (b) between high-level human readable host/node identifiers to network layer addresses (e.g. Fully Qualified

Domain Names (FQDN) to IP addresses, a function implemented by a Domain Name System (DNS) server), and (c) translation between node/service identifiers and network locators in case the higher layers above the networking layer use node or service identifiers that are decoupled from the node addresses in the network (e.g.Host Identity Protocol (HIP; Moskovitz & Nikander 2006) identifiers and IP addresses).

- Potential fragmentation and reassembly of messages due to limitations of the underlying layers is also handled by the Network FC.
- Finally, the Network FC is responsible for handling messages that cross different networking or MAC/PHY layer technologies, a function that is typically implementedon a network gateway type of device.
- The End-to-End Communication FC is responsible for end-to-end transport of application layer messages through diverse network and MAC/PHY layers.
- In turn, this means that it may be responsible for end-to-end retransmissions of missing frames depending on the configuration of the FC. For example, if the End-to- End Communication FC is mapped in an actual system to a component implementingthe Transmission Control Protocol (TCP) protocol, reliable transfer of frames dictates the retransmission of missing frames.
- Finally, this FC is responsible for hosting any necessary proxy/cache and any protocol translation between networks with different transport/application layer technologies. An example of such functionality is the HTTP-CoAP proxy, which performs transport-layer protocol translation. The End-to-End FC interfaces the Network FCon the "southbound" direction.

IoT Service functional group

The IoT Service FG consists of two FCs: The IoT Service FC and the IoT ServiceResolution FC:

- The IoT Service FC is a collection of service implementations, which interface the related and associated Resources. For a Sensor type of a Resource, the IoT Service FC includes Services that receive requests from a User and returns the Sensor Resource value in synchronous or asynchronous (e.g. subscription/notification) fashion.
- The services corresponding to Actuator Resources receive User requests for actuation, control the Actuator Resource, and may return the status of the Actuator after the action.
- A Tag IoT Service can behave both as a Sensor (for reading the identifier of the Tag), or as an Actuator (for writing a new identifier or information on the Tag, if possible).
- The IoT Service Resolution FC contains the necessary functions to realize a directory of IoT Services that allows dynamic management of IoT Service descriptions and discovery/lookup/resolution of IoT Services by other Active Digital Artifacts.
- The Service descriptions of IoT Services contain a number of attributes as seen earlier in the IoT Functional Model section. Dynamic management includes methods such as creation/update/ deletion (CUD) of Service description, and can be invoked by both the
- IoT Services themselves, or functions from the Management FG (e.g.bulk creation of IoT Service descriptions upon system start-up).

- The discovery/lookup and resolution functions allow other Services or Active Digital Artifacts to locate IoT Services by providing different types of information to the IoT Service Resolution FC.
- By providing the Service identifier (attribute of the Service description) a lookup method invocation to the IoT Service Resolution returns the Service description, while the resolution method invocation returns the contact information (attribute of the service description) of a service for direct Service invocation (e.g. URL).
- The discovery method, on the other hand, assumes that the Service identifier is unknown, and the discovery request contains a set of desirable Service description attributes that matching Service descriptions should contain.

Virtual Entity functional group

- The **Virtual Entity FG** contains functions that support the interactions between Users and Physical Things through Virtual Entity services.
- An example of such an interaction is the query to an IoT system of the form, "What is the temperature in the conference room Titan?" The Virtual Entity is the conference room "Titan," and the conference room attribute of interest is "temperature."
- Assuming that the room is actually instrumented with a temperature sensor, if the User had the knowledge of which temperature sensor is installed in the room (e.g. TempSensor #23), then the User could re-formulate and re-target this query to, "What is the value of TempSensor #23?" dispatched to the relevant IoT Service representing the temperature resource on the TempSensor #23.
- The Virtual Entity interaction paradigm requires functionality such as discovery of IoT Services based on Virtual Entity descriptions, managing the Virtual Entity-IoT Service associations, and processing Virtual Entity-based queries. The following FCs are defined for realizing these functionalities:
- The **Virtual Entity Service FC** enables the interaction between Users and Virtual Entities by means of reading and writing the Virtual Entity attributes (simple or complex), which can be read or written, of course.
- Some attributes (e.g. the GPS coordinates of a room) are static and non-writable by nature, and some other attributes are non-writable by access control rules.
- In general attributes that are associated with IoT Services, which in turn represent Sensor Resources, can only be read. There can be, of course, special Virtual Entities associated with the same Sensor Resource through another IoT Service that allow write operations.
- An example of such a special case is when the Virtual Entity represents the Sensor device itself (for management purposes).
- In general, attributes that are associated with IoT Services, which in turn represent Actuator Resources, can be read and written. A read operation returns the actuator status, while a write operation results in a command sent to the actuator.
- The **Virtual Entity Registry FC** maintains the Virtual Entities of interest for the specific IoT system and their associations.

- The component offers services such as creating/reading/updating/deleting Virtual Entity descriptions and associations. Certain associations can be static; for example, the entity “Room #123” is contained in the entity “Floor #7” by construction, while other associations are dynamic, e.g. entity “Dog” and entity “Living Room” due to at least Entity mobility. Update and Deletion operations take the Virtual Entity identifieras a parameter.
- The Virtual Entity Resolution FC maintains the associations between Virtual Entities and IoT Services, and offers services such as creating/reading/updating/deleting associations as well as lookup and discovery of associations.
- The Virtual Entity Resolution FC also provides notification to Users about the status of the dynamic associations between a Virtual Entity and an IoT Service, and finally allows the discovery of IoT Services provided the certain Virtual Entity attributes.
- The Virtual Entity and IoT Service Monitoring FC includes: (a) functionality to assert static Virtual EntityIoT Service associations, (b) functionality to discover new associations based on existing associations or
- Virtual Entity attributes such as location or proximity, and (c) continuous monitoring of the dynamic associations between Virtual Entities and IoT Services and updates of their status in case existing associations are not valid any more.

IoT process management functional group

- The IoT Process Management FG aims at supporting the integration of business processes with IoT-related services. It consists of two FCs:

✓ The Process Modeling FC provides that right tools for modeling a business process that utilizes IoT-related services.

✓ The Process Execution FC contains the execution environment of the process models created by the Process Modelling FC and executes the created processes by utilizing the Service Organization FG in order to resolve high-level application requirements tospecific IoT services.

✓ **Service Organization functional group**

✓ The Service Organization FG acts as a coordinator between different Services offered by the system. It consists of the following FCs:

- The Service Composition FC manages the descriptions and execution environment of complex services consisting of simpler dependent services. An example of a complex composed service is a service offering the average of the values coming from a number of simple Sensor Services. The complex composed service descriptions can be wellspecified or dynamic/flexible depending on whether the constituent services are well-defined and known at the execution time or discovered on-demand. The objective of a dynamic composed service can be the maximization of the quality of information achieved by the composition of simpler Services, as is the case with the example “average” service described earlier.

- The Service Orchestration FC resolves the requests coming from IoT Process Execution FC or User into the concrete IoT services that fulfill the requirements.

- The Service Choreography FC is a broker for facilitating communication among Services using the Publish/Subscribe pattern. Users and Services interested in specific IoT- related services subscribe to the Choreography FC, providing the desirable service attributes even if the desired services do not exist. The Choreography FC notifies the Users when services fulfilling the subscription criteria are found.

✓ **Security functional group**

✓ The Security FG contains the necessary functions for ensuring the security and privacy of an IoT system. It consists of the following FCs:

- The Identity Management FC manages the different identities of the involved Services or Users in an IoT system in order to achieve anonymity by the use of multiple pseudonyms.

- The Authentication FC verifies the identity of a User and creates an assertion upon successful verification. It also verifies the validity of a given assertion.

- The Authorization FC manages and enforces access control policies. It provides services to manage policies (CUD), as well as taking decisions and enforcing them regarding access rights of restricted resources. The term "resource" here is used as a representation of any item in an IoT system that needs a restricted access. Such an item can be a database entry (Passive Digital Artifact), a Service interface, a Virtual Entity attribute (simple or complex), aResource/Service/Virtual Entity description, etc.

✓ The Key Exchange & Management is used for setting up the necessary security keys between two communicating entities in an IoT system.

✓ The Trust & Reputation FC manages reputation scores of different interacting entities in an IoT system and calculates the service trust levels.

✓ **Management functional group**

✓ The Management FG contains system-wide management functions that may use individual FC management interfaces. It is not responsible for the management of each component, rather for the management of the system as

✓ a whole. It consists of the following FCs:

- The Configuration FC maintains the configuration of the FCs and the Devices in an IoT system (a subset of the ones included in the Functional View). The component collects the current configuration of all the FCs and devices, stores it in a historical database, and compares current and historical configurations. The component can also set the system-wide configuration (e.g. upon initialization), which in turn translates to configuration changes to individual FCs and devices.

✓ The Fault FC detects, logs, isolates, and corrects system-wide faults if possible. This means that individual component fault reporting triggers fault diagnosis and fault recovery procedures in the Fault FC.The Member FC manages membership information about the relevant entities in an IoT system. Example relevant entities are the FGs, FCs Services, Resources, Devices, Users, and Applications. Membership

✓ information is typically stored in a database along with other useful information such as capabilities, ownership,

and access rules & rights, which are used by the Identity Management and Authorization FCs.

- The State FC is similar to the Configuration FC, and collects and logs state information from the current FCs, which can be used for fault diagnosis, performance analysis and prediction, as well as billing

✓ purposes. This component can also set the state of the other FCs based on system-wise state information.

- The Reporting FC is responsible for producing compressed reports about the system state based on input from FCs.

Information view

- The information view consists of (a) the description of the information handled in the IoT System, and (b) the way this information is handled in the system; in other words, the information lifecycle and flow (how information is created, processed, and deleted), and the information handling components.

- **Information description**

The pieces of information handled by an IoT system complying to an ARM such asthe IoT-A (Carrez et al. 2013) are the following:

- Virtual Entity context information, i.e. the attributes (simple or complex) as represented by parts of the IoT Information model (attributes that have values and metadata such as the temperature of a room). This is one

of the most important pieces of information that should be captured by an IoT system,and represents the properties of the associated Physical Entities or Things.

- IoT Service output itself is another important part of information generated by an IoT system. For example, this is the information generated by interrogating a Sensor or a Tag Service.

- Virtual Entity descriptions in general, which contain not only the attributes coming from IoT Devices (e.g. ownership information).

- Associations between Virtual Entities and related IoT Services.

- Virtual Entity Associations with other Virtual Entities (e.g. Room #123 is on Floor #7).
- IoT Service Descriptions, which contain associated Resources, interface descriptions,etc.

- Resource Descriptions, which contain the type of resource (e.g. sensor), identity, associated Services, and Devices.

- Device Descriptions such as device capabilities (e.g. sensors, radios).

- Descriptions of Composed Services, which contain the model of how a complex service is composed of simpler services.

- IoT Business Process Model, which describes the steps of a business process utilizing other IoT-related services (IoT, Virtual Entity,Composed Services).

- Security information such as keys, identity pools, policies, trust models, reputation scores, etc.
- Management information such as state information from operational FCs used for fault/performance purposes, configuration snapshots, reports, membership information, etc.

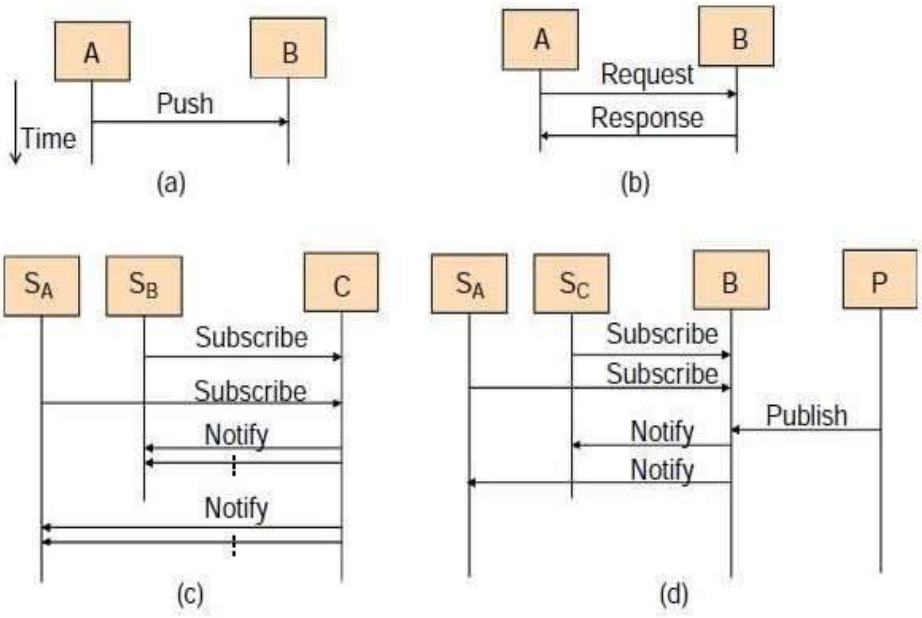


Fig. 2.7 Information exchange patterns.

- **Push:** An FC A pushes the information to another FC B provided that the contact information of the component B is already configured in component A, and component B listens for such information pushes.
- **Request/Response:** An FC A sends a request to another FC B and receives a response from B after A serves the request. Typically the interaction is synchronous inthe sense that A must wait for a response from B before proceeding to other tasks, but in practice this limitation can be realized with parts of component A waiting, and other parts performing other tasks. Component B may need to handle concurrent requests and responses from multiple components, which imposes certain requirements on the capabilities for the device or the network that hosts the FC.
- **Subscribe/Notify:** Multiple subscriber components (SA, SB) can subscribe for information to a component C, and C will notify the relevant subscribers when the requested information is ready. This is typically an asynchronous information request after which each subscriber can perform other tasks. Nevertheless, a subscriber needs to have some listening components for receiving the asynchronous response. The target component C also needs to maintain state information about which subscribers requested which information and their contact information.
- The Subscribe/Notify pattern is applicable when typically one component is the hostof the information needed by multiple other components. Then the subscribers need only establish a Subscribe/Notify relationship with one component. If multiple components can be information producers or information hosts, the Publish/Subscribe pattern is a more scalable solution from the point of view of the subscribers.
- **Publish/Subscribe:** In the Publish/Subscribe (also known as a Pub/Sub pattern), there is a third component called the broker B, which mediates subscription and publications between subscribers (information

consumers) and publishers (or information producers). Subscribers such as SA and SB subscribe to the broker about the information they are interested in by describing the different properties of the information. Publishers publish information and metadata to the broker, and the broker pushes the published information to (notification) the subscribers whose interests match the published information.

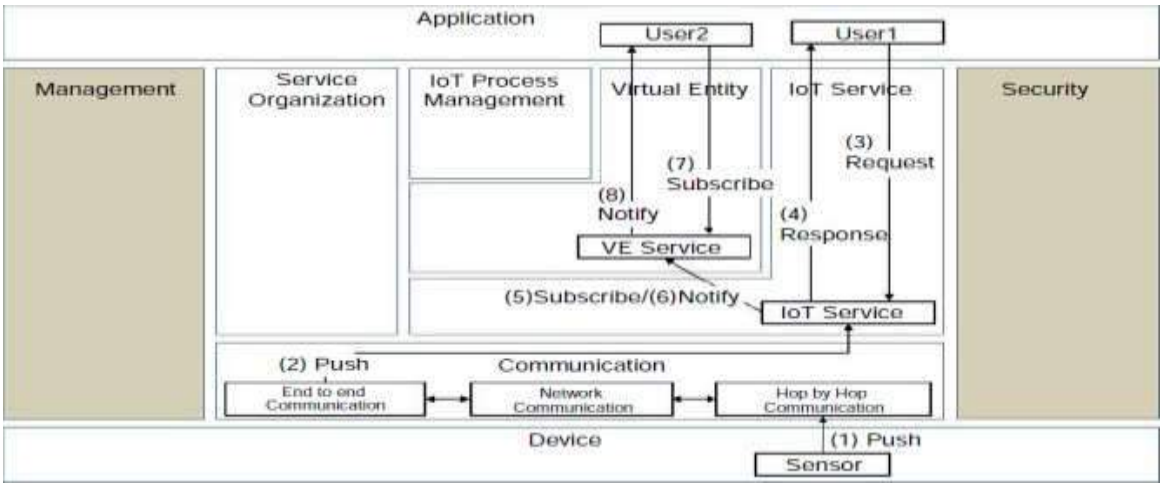


Fig. 2.8 Device, IoT Service, and Virtual Entity Service Interactions.

- In Figure it is assumed that the generated sensed data is pushed by a sensor device (under Steps 1 and 2) that is part of a multi-hop mesh network such as IEEE 802.15.4 through the Hop-by-Hop, Network, and End-to-End communication FCs towards the Sensor Resource hosted in the network.

- Please note that the Sensor Resource is not shown in the figure, only the associated IoT Service. A cached version of the sensor reading on the Device is maintained onthe IoT Service. When User1 (Step 3) requests the sensor reading value from the specific Sensor Device (assuming User1 provides the Sensor resource identifier), the IoT Service provides the cached copy of the sensor reading back to the User1 annotated with the appropriate metadata information about the sensor measurement, for example, timestamp of the last known reading of the sensor, units, and location of the Sensor Device.
- Also assume that that the Virtual Entity Service associated with the Physical Entity (e.g. a room in a building) where the specific Sensor Device has been deployed already contains the IoT Service as a provider of the "hasTemperature" attribute of its description. The Virtual Entity Service subscribes to the IoT Service for updates of the sensor readings pushed by the Sensor Device (Step 5). Every time the Sensor Device pushes sensor readings to the IoT Service, the IoT Service notifies (Step 6) the Virtual Entity Service, whichupdates the value of the attribute "hasTemperature" with the sensor reading of the Sensor Device. At a later stage, a User2 subscribing (Step 7) to changes on the VirtualEntity attribute "hasTemperature" is notified every time the attribute changes value (Step 8).

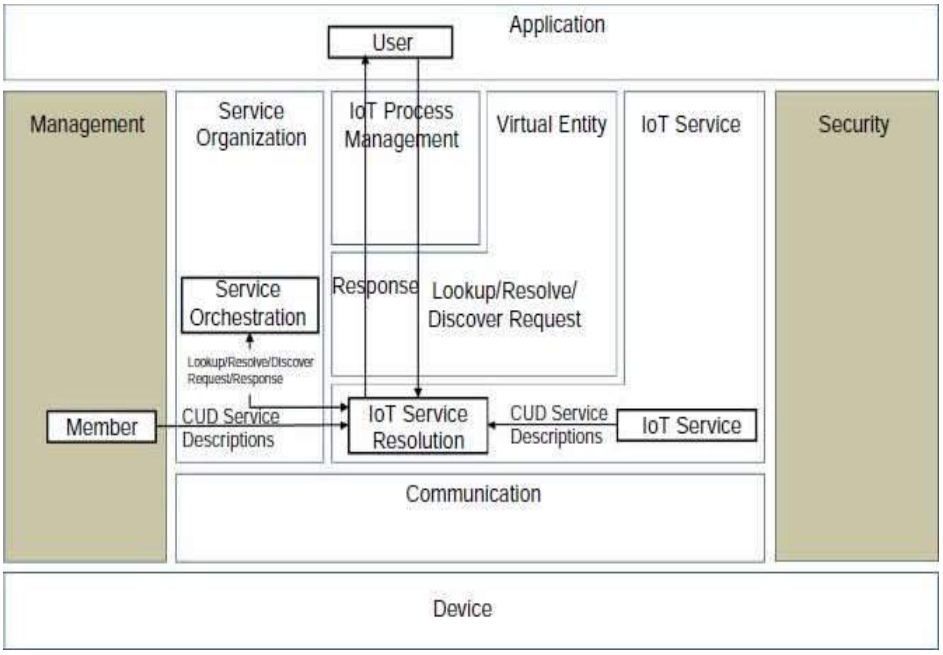


Fig. 2.9 IoT Service Resolution

- Figure depicts the information flow when utilizing the IoT Service Resolution FC. The IoT Service Resolution implements two main interfaces, one for the CUD of Service Description objects in the IoT Service Resolution database/store, and one for lookup/resolution/discovery of IoT Services.
- As a reminder, the lookup and resolution operations provide the Service Description and the Service locator, respectively, given the Service identifier and the discovery operation returns a (set of) Service Description(s) given a list of desirable attributes that matching Service Descriptions should contain.
- The CUD operations can be performed by the IoT Service logic itself or by a management component (e.g. Member FC in Figure). The lookup/resolution and discovery operation can be performed by a User as a standalone query or the Service Orchestration as a part of a Composed Service or an IoT Process.
- If a discovery operation returns multiple matching Service Descriptions, it is upon the User or the Service Orchestration component to select the most appropriate IoTService for the specific task.
- Although the interactions in Figure follow the Request/Response patterns, the lookup/resolution/discovery operations can follow the Subscribe/Notify pattern in the sense that a User or the Service Orchestration FC subscribe to changes of existing IoT Services for lookup/resolution and for the discovery of new Service Descriptions in the case of a discovery operation.

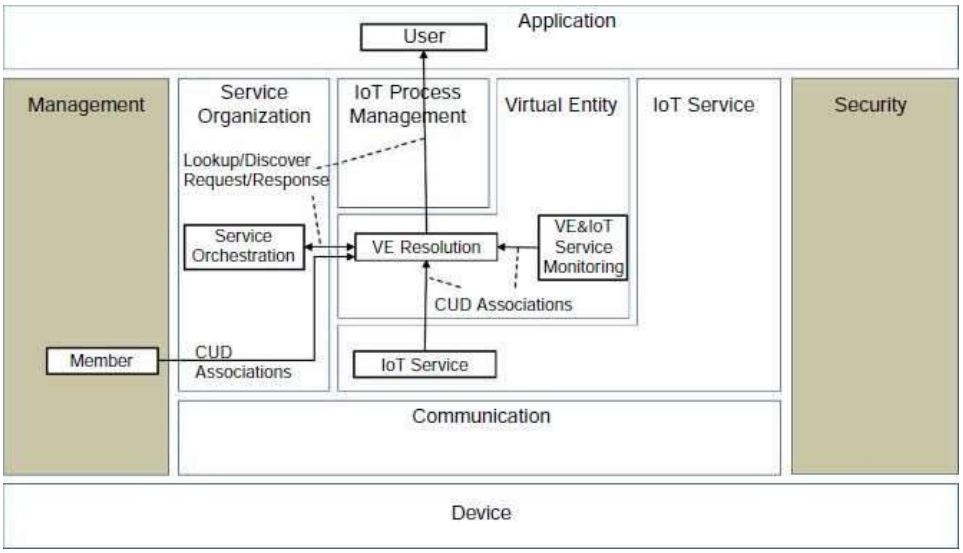


Fig. 2.10 Virtual Entity Service Resolution

- Figure describes the information flow when the Virtual Entity Service Resolution FC is utilized. The Virtual Entity Resolution FC allows the CUD of Virtual Entity Descriptions, and the lookup and discovery of Virtual Entity Descriptions.
- A lookup operation by a User or the Service Orchestration FC returns the Virtual Entity Description given the Virtual Entity identity, while the discovery operation returns the Virtual Entity Description(s) given a set of Virtual Entity attributes (simpleor complex) that matching Virtual Entities should contain.
- Note that the Virtual Entity Registry is also involved in the information flow because it is the storage component of Virtual Entity Descriptions, but it is omitted from the figure to avoid cluttering. The Virtual Entity Resolution FC mediates the requests/responses/ subscriptions/notifications between Users and the Virtual Entity Registry, which has a simple create/read/update/delete (CRUD) interface given the Virtual Entity identity.
- The FCs that could perform CUD operations on the Virtual Entity Resolution FC are the IoT Services themselves due to internal configuration, the Member Management FC that maintains the associations as part of the system setup, and the Virtual Entity and IoT Service Monitoring component whose purpose is to discover dynamic associations between Virtual Entities and IoT Services.
- It is important to note that the Subscribe/Notify interaction patterns can also be applicable to the lookup/ discovery operations, the same as the Request/Response patterns provided the involved FCs implement Subscribe/Notify interfaces.

Deployment and operational view

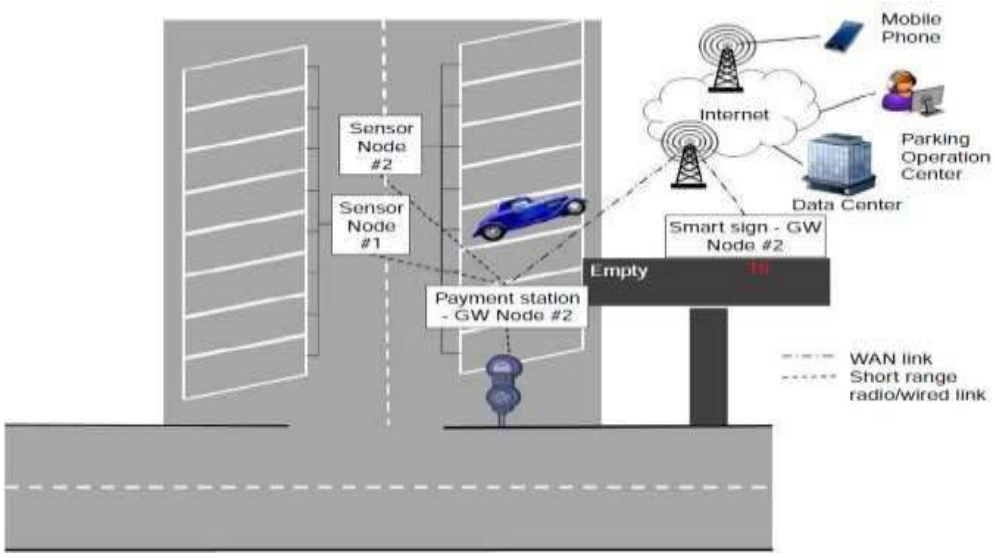


Fig. 2.11 Parking Lot Deployment and Operational View, Devices.

- The Deployment and Operational View depends on the specific actual use case and requirements, and therefore we present here one way of realizing the Parking Lot example seen earlier.
- Figure depicts the Devices view as Physical Entities deployed in the parking lot, as well as the occupancy sign. There are two sensor nodes (#1 and #2), each of which are connected to eight metal/car presence sensors.
- The two sensor nodes are connected to the payment station through wireless or wired communication. The payment station acts both as a user interface for the driver to pay and get a payment receipt as well as a communication gateway that connects the two sensor nodes and the payment interface physical devices (displays, credit card slots, coin/note input/output, etc.) with the Internet through Wide Area Network (WAN) technology.
- The occupancy sign also acts as a communication gateway for the actuator node (display of free parking spots), and we assume that because of the deployment, a direct connection to the payment station is not feasible (e.g. wired connectivity is too prohibitive to be deployed or sensitive to vandalism).
- The physical gateway devices connect through a WAN technology to the Internet and towards a data center where the parking lot management system software is hosted as one of the virtual machines on a Platform as a Service (PaaS;) configuration.
- The two main applications connected to this management system are human user mobile phone applications and parking operation center applications. We assume that the parking operation center manages several other parking lots using similar physicaland virtual infrastructure.

- Figure shows two views superimposed, the deployment and functional views, for the parking lot example. Please note that several FGs and FCs are omitted here for simplicity purposes, and certain non-IoT-specific

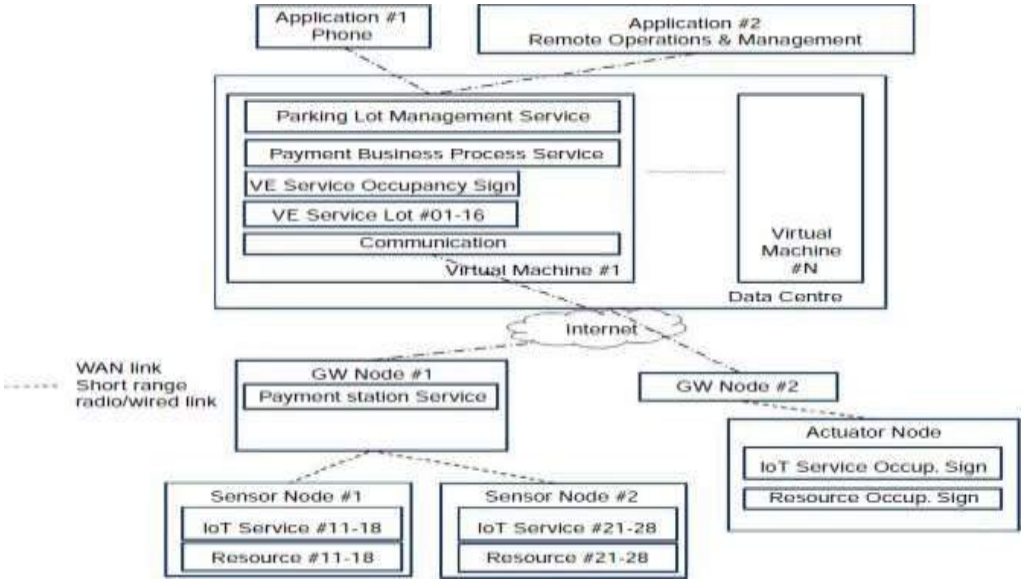
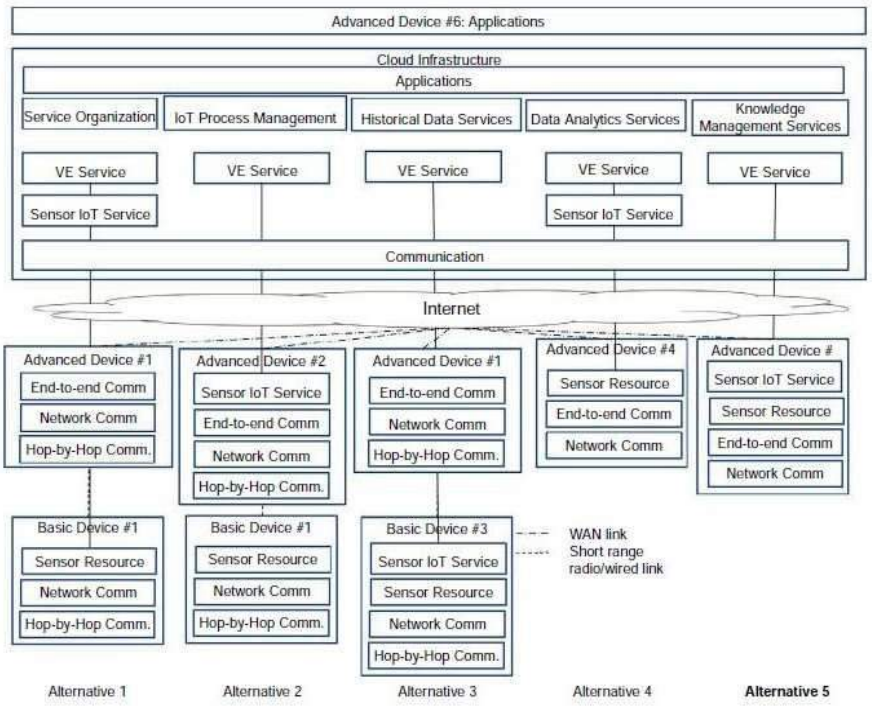


Fig 2.11 Parking Lot Deployment & Operational View, Resources, Services, VirtualEntities, Users

- Services appear in the figure 5.7 because an IoT system is typically part of a larger system. Starting from the Sensor Devices, as seen earlier, Sensor Node #1 hosts Resource #11#18, representing the sensors for the parking spots #01#08, while earlier Sensor Node #2 hosts Resource #21#28, representing the sensors for the parking spots#09#16.

- Assume that the sensor nodes are powerful enough to host the IoT Services #11#18 and #21#28 representing the respective resources. The two sensor nodes are connected to the gateway device that also hosts the payment service with the accompanying sensors and actuators, as seen earlier. The other gateway device hosts the occupancy sign actuator resource and corresponding service. The management system for the specific parking lot, as well as others, is deployed on a virtual machine on a data center. The virtual machine hosts communication capabilities, Virtual Entity services for the parking spots #01#16, the Virtual Entity services for the occupancy sign, a payment business process that involves the payment station and input from the occupancy sensor services, and the parking lot management service that provides exposure and access control to the parking lot occupancy data for the parking operation center and the

- consumer phone applications.As a reminder, the Virtual Entity service of the parking lot uses the IoT Services hosted on two sensor nodes and performs the mapping between the sensor node identifiers (#11#18 and #21#28) to parking spot identifiers (spot #01#16). The services offered on these parking spots are to read the current state of the parking spot to see whether it is “free” or “occupied.” The Virtual Entity corresponding to the occupancy sign contains one writable attribute: the number of free parking spots. A User writing this Virtual



- **Fig 2.12 Mapping IoT Domain Model concepts to Deployment View**

- Entity attribute results in an actuator command to the real actuator resource to change its display to the new value.

- Figure shows an example of mapping an IoT Domain Model and Functional View to Devices with different capabilities (different alternatives) connecting to a cloud infrastructure. Alternative 1 shows devices that can host only a simple Sensor Device and a short-range wired or wireless connectivity technology (Basic Device #1).

  • Such kind of device needs an Advanced Device of type #1 that allows the basic device to perform protocol adaptation (at least from the short-range wired or wireless connectivity technology to a WAN technology) so that the Sensor IoT service in thecloud and the Sensor Resource on the Basic Device #1 can exchange information.

- The Virtual Entity representing the Physical Entity where the Basic Device #1 is deployed is also hosted in the cloud.

- In alternative 2, Advanced Devices (type #2) can host the Sensor IoT Service communicating to the Sensor Resource on a Basic Device #1.

- The cloud infrastructure in this case only hosts the Virtual Entity Service corresponding to the Sensor IoT Service. The difference between alternative 1 and 2 is that the Sensor IoT Service hosted on an Advanced Device #2 should be capable of responding to requests from Users (cloud services, Applications) with the

appropriate secure mediation of course.

- In alternative 3, the Basic Device #3 is capable of providing the Sensor Resource and the Sensor IoT Service but still needs an Advanced Device #1 to transport IoT servicerequests/responses/subscriptions/
- notifications/publications to the Users in the cloud. According to experience, this kind of deployment scenario imposes a high burden on a Basic Device, which potentially makes the Basic Device the weakest link in the information flow
- If malicious Users launch a Denial of Service (DoS) attack on the node, the probability of the node going down is very high.
- Alternatives 4 and 5 show Advanced Devices offering a WAN interface.In alternative 4, only the Sensor Resource is hosted on the Device, while in alternative 5, even the IoT Service is hosted on the Device. The Virtual Entity Service is hosted in the cloud.

Real-World Design Constraints

Devices and Networks:

- The devices that form networks in the M2M Area Network domain must be selected, or designed, with certain functionality suitable to IoT applications.
- The devices must have an energy source (e.g. batteries), computational capability (e.g. an MCU), appropriate communications interface (e.g. a Radio Frequency Integrated Circuit (RFIC) and front end RF circuitry), memory (program and data), and sensing (and/or actuation) capability.
- These must be integrated in such a way that the functional requirements of the desired application can be satisfied with additional nonfunctional requirements.

Functional Requirements:

1. Specific sensing and actuating capabilities
2. Sensing principle and data requirements: Sometimes continuous sampling of sensing data is required. For some applications, sampling after specific intervals is required.
3. The parameters like higher network throughput, data loss, energy use, etc are decided based on sensing principle.

Sensing and communications field:

- The sensing field is to be considered for sensing in local area or distributed sensing. The distance between sensing points is also important factor to be considered.
- The physical environment has an implication on the communications technologies selected and the reliability of the system in operation thereafter.
- Devices must be placed in close enough proximity to communicate. Where the distance is too great, routing devices may be necessary.

Programming and embedded intelligence:

- Devices in the IoT are heterogeneous such as various computational architectures, including MCUs (8-, 16-,32-bit, ARM, 8051, RISC, Intel, etc.), signal conditioning (e.g. ADC), and memory (ROM, S/F/D) RAM, etc.), communications media, peripheral components (sensors, actuators, buttons, screens, LEDs), etc.
- In every case, an application programmer must consider the hardware selected or designed, and its capabilities.

- Application-level logic decides the sampling rate of the sensor, the local processing performed on sensor readings, the transmission schedule (or reporting rate), and the management of the communications protocol stack, among other things.
- The programmers have to reconfigure and reprogram devices in case of change in devices in IoT application.

Power:

- Power is essential for any embedded or IoT device.
- Depending on the application, power may be provided by the mains, batteries, or hybrid power sources.
- Power requirements of the application are modeled prior to deployment. This allows the designer to estimate the cost of maintenance over time.

Gateway:

- Gateway devices or proxies are selected according to need of data transitions.

Nonfunctional requirements:

The non-functional requirements are technical and non-technical.

1. Regulations:

- For applications that require placing nodes in public places, prior permissions are important.
- Radio Frequency (RF) regulations limit the power with which transmitters can broadcast.

2. Ease of use, installation, maintenance, accessibility:

- This relates to positioning, placement, site surveying, programming, and physical accessibility of devices for maintenance purposes.

3. Physical constraints:

- Integration of additional electronics into existing system
- Suitable packaging
- Kind and size of antenna
- Type of power supply

Financial cost:

Financial cost considerations are as follows:

- Component Selection: Typically, the use of these devices in the M2M Area Network domain is to reduce the overall cost burden. However, there are research and development costs likely to be incurred for each individual application in the IoT that requires device development or integration. Developing devices in small quantities is expensive.
- Integrated Device Design: Once the energy, sensors, actuators, computation, memory, power, connectivity, physical, and other functional and nonfunctional requirements are considered, it is likely that an integrated device must be produced.

Data representation and visualization:

Each IoT application has an optimal visual representation of the data and the system. Data that is generated from heterogeneous systems has heterogeneous visualization requirements. There are currently no satisfactory standard data representation and storage methods that satisfy all of the potential IoT applications.

- **Data Representation and Visualization**

- IoT Data Visualization is the technique where the raw data is presented into a more insightful one that is derived from different data streams. It analysis the data and looks into the certain patterns & behaviours that

improves with better business decision making. It helps to create a viable business strategy.

- The Data Visualization Helps to Unlock Multiple Insightful Values
- Helps to make real-time decisions with the combination of multiple data sources into a single insightful dashboard with multi-layered visual data.
- Combines the new IoT data transmitted from data sensors with the existing data to analyse and bring light to new business opportunities.
- Supports to monitor IoT devices and infrastructure for better performance on IoT dataflow.
- Helps to analyse multiple data correlations in real-time.
- Data Visualization Tools for IoT application:
- **Grafana Tool:**
- Grafana supports various data sources seamlessly like Elasticsearch, MySQL,PostgreSQL, Graphite, Prometheus and so on.
- Provides time series analytics to monitor, analyze data over a period of time.
- Upbeat of this Grafana tool is it provides on-premises cloud storage or any othercloud of your choice, which gives complete control of the infrastructure.
- Alert notification can be set up whenever an unfavourable event occurs which getsprompt notification using any communication platform.
- It has several built-in support features like Graphite, CloudWatch, Elastic Search,InfluxDB.
- Kibana Tool is an open source data visualization tool for analyzing large volumes of log data. To work with Kibana tool, it needs two more technological stack which is Elasticsearch and Logstash. It is popularly known as ELK stack, globally used log management platform.
- **Kibana Tool:**
- Working of kibana
- Initially, the logstash is responsible to collect all the data from the various remote sources
- Next, these data logs are then pushed and sent to the Elasticsearch
- Elasticsearch acts as the database to the kibana tool with all the log information
- Finally, Kibana tool presents these log data in the form of pie charts, bar or linegraphs to the user.
- Highlights of Kibana:
- Canvas visualization gives colorful visual data comprising of different patterns, texts known as workpad. Kibana also represents data in the form of bar chart, pie chart, heat map, line graph and so on.

- Contains Interactive dashboards and easily it can be converted into reports for future references
- Create visualization with the help of several dev tools where you can work with indexes to add, delete and update the data.
- Timelion, a timeline visualization tool helps to get the historical data and compare them with current data for getting deeper analysis.
- Supports third-party plugins and to get near to real experience view, it effectively uses coordinate and region maps
- Power BI Tool for Real-Time Data Visualization
- Microsoft's product PowerBI is a popular Business Intelligence Tool. Like its predecessors, Tableau and other BI tools, it provides a detailed analysis reports for large Enterprises. Power BI comes with a suite of products with Power BI desktop, mobile Power BI apps and Power BI services for SaaS.
- Power BI Desktop – Helps to create reports
- Power BI Services – Helps to Publish those reports
- Power BI mobile app – Help to Views the reports and dashboards
- How does Power BI work?
- First, the data is collected from the external data sources. With 'Get Data' option it allows you to get information from various sources including Facebook, Google Analytics, Azure Cloud, Salesforce etc. Also, it provides ODBC connection to get ODBC data as well.
- Using Power BI, you can create visualization in 2 ways, one is by adding from the right-side panel to the report canvas which is in a table type visualization format or by simple drag and drop of value axis under visualization. Once the report is developed,it can be published to web portal with the help of Power BI service. We can access thereport, export it in pdf, excel or any preferred format.
- **Highlights of Power BI:**
- Though PowerBI offers paid services, it is comparatively cheaper than other BI tools. It offers free services upto 1GB storage
- Helps to analyse both streaming and static data
- Provides rich data visualization
- Short learning curve
- Provides IoT integrations
- **Industrial Automation**
- Industrial automation is all about intelligent process control. The IoT doesn't depend on your hardware because you can choose independent control systems, sensors, and network components. The IoT's power goes beyond

the limited features and functionalities your device manufacturer or software provider offers. Those who use the IoT in industrial automation processes can connect multiple sites and locations so that they operate in harmony. Industrial automation is well-known for diverting technology from the commercial sphere and adapting it to new ends. The industry's widespread IoT adoption builds upon this tested concept in numerous ways:

- Wireless Improvement of Existing Monitoring and Control Systems
- Wireless connectivity makes it simpler to implement complex control systems in awkward, remote or hazardous environments. For instance, using wired networks to link remotely controlled cranes, robot arms and other manufacturing devices can be problematic due to their unique ranges of motion and exposure to harsh fabrication environments. The IoT's compatibility with wireless technology lets enterprises replace standard linkages with fully enclosed mesh radios that perform the same functions. Even better, these alternatives may be more useful for automation processes that require fine-tuning or ongoing adjustments. For instance, you don't have to replace miles of Ethernet cable to achieve higher transmission speeds with Wi-Fi.
- Building Factories That Build and Run Themselves
- Growth has decided on the pros and cons. Although few experiences beat the thrill of taking your organizational training wheels off and cruising along, doing business at a higher volume introduces unique risks, such as the potential for greater waste should you take a wrong turn. A company that wanted to conserve resources might use an industrial sensor system to tell it when to shut down auxiliary production lines. An enterprise that relies on automated stock machines to transport replacement parts to workstations could employ a connected framework to initiate new deliveries without waiting for approval from a line manager. The IoT also makes it possible to create digital twins. These replicas of existing systems serve as testbeds for new projects andexperiments.
- Managing Communications Whenever the Need Arises
- The IoT enhances traditional automation schemes by making everything on-demand. When you make a change from a control dashboard, you get to see its effects ripple outward right away. What you might not expect is that the system also performs the innumerable tedious tasks that facilitate good digital communication, such as
- • Rerouting traffic to keep data moving no matter how much information happens to be passing through,
- • Accounting for the effects of network topologies to sustain optimized service quality,
- • Accommodating vendor-neutral communication protocols and schemes to support a wider variety of hardware and software,
- • Self-detecting equipment failures and automatically switching to functional network elements, and
- • Duplicating and storing data as necessary to prevent catastrophic losses.
- Although this kind of work may get overlooked because it goes on in the background, it's an essential part of ensuring that automation frameworks behave deterministically. When your industrial communication systems behave consistently, soundmanagement practices prove easier to execute.
- Decentralizing Debugging and Maintenance

- There’s no shortage of industrial automation maintenance philosophies to choose from, so debugging can get confusing. IoT mesh networks help stakeholders handle maintenance more logically. You can debug, tweak and maintain controllers and sensors from local network nodes to cut down on overhead and make the best use of limited bandwidth. Decentralized maintenance is the glue that helps automation systems stick together and run seamlessly even as they expand. By using the IoT to program functions at the node level, you can optimize resource usage and slash costs for a more productive enterprise.

- Investing in the IoT in Industrial Automation Settings

- Internet of Things technologies offer a spectrum of other potential benefits that we haven’t even covered. There are voice-recognition systems that let factory owners authenticate themselves and implement complex behaviours without any manual programming. Embedded and linked networks contribute to improved lifecycle oversight, demand-specific customization and better cost control, but choosing the best-equipped IoT layout and technical components can be a tough task. Optimality isn’t universal. It’s defined by the circumstances, so you need to move forward with an eye on building something that’s sufficiently flexible yet robust enough to survive the unexpected.

- **Interaction and Remote control**

- IoT devices produce many types of information, including telemetry, metadata, state, and commands and responses. Telemetry data from devices can be used in short operational timeframes or for longer-term analytics and model building. (For more on this diversity, read the overview of Internet of Things.)
    - Many devices support local monitoring in the form of a buzzer or an alarm panel on- premises.

  This type of monitoring is valuable, but has limited scope for in-depth or long-term analysis. This article instead discusses remote monitoring, which involves gathering and analysing monitoring information from a remote location using cloudresources. Operational and device performance data is often in the form of a time series, where each piece of information includes a time stamp. This data can be further enriched with dimensional labels (sometimes referred to as tags), such as labels that identify hardware revision, operating time zone, installation location, firmware version, and so on.

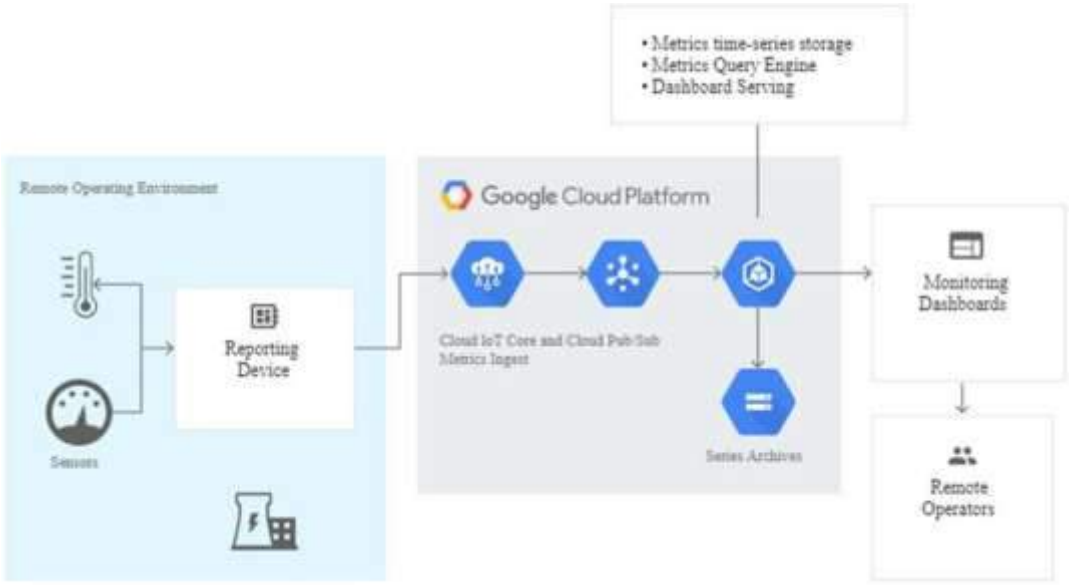
- Time-series telemetry can be collected and used for monitoring. Monitoring in this context refers to using a suite of tools and processes that help detect, debug, and resolve problems that occur in systems while those systems are operating. Monitoring can also give you insight into the systems and help improve them.

- The state of monitoring IT systems, including servers and services, has continuously improved. Monitoring tools and practices in the cloud-native world of microservices and Kubernetes are excellent at monitoring based on time-series metric data. These tools aren't designed specifically for monitoring IoT devices or physical processes, but the constituent parts—labelled series of metrics, visualization, and alerts—all can apply to IoT monitoring.

- **Remoteness**

- Unlike servers in a cluster, monitored devices might be far from the systems that are organizing the metric data and providing visualizations. There is debate in the monitoring community about push-based versus pull-based collection methods for monitoring telemetry. For IoT devices, push-based monitoring can be moreconvenient. But you must consider the trade-offs in the entire stack (including things like the power of the query language, and the efficiency and cost of the time-series storage) when you choose which metrics framework to use.

- In either approach, a remote device might become disconnected from the monitoring system. No effective monitoring can occur if data isn't flowing. Stale and missing metrics can hamper the value of a metric series where you might be calculating rates or other types of values derived over time. When you're monitoring remote devices, it's also important to recognize that variation in timestamps is possible and to ensure the best clock synchronization possible. The following diagram shows a schematic of remote devices, with centralized monitoring compared to cluster-based monitoring.



- **Fig 2.13 Remote devices with centralized monitoring**

- **Monitoring design patterns**

- When it is determined which systems you're monitoring, you need to think about why you're monitoring. The system you're working with is providing a useful function, and the goal of monitoring is to help ensure that a function or service is performing as intended.

- When monitoring software services, you look for measurements around the performance of that service, such as web request response times. When the service is a physical process such as space heating, electrical generation, or water filtration, you might use devices to instrument that physical process and take measurements of things like engine hours or cycle times. Whether you're using a device as a means solely to instrument a physical process, or whether the device itself is performing a service, you want to have a number of measurements about the device itself.

- Measurements made at the point of instrumentation result in a metric being sent and recorded in the centralized monitoring system. Metrics might be low level (direct and unprocessed) or high level (abstract). Higher-level metrics might be computed from lower-level metrics. One should start by thinking about the high-level metrics you need in order to ensure delivery of service. One can then determine which lower-level metrics you need to collect in order to support your monitoring goals. Not all metrics are useful, and it's important not to fall into

the trap of measuring things just because you can, or because they look impressive (so called "vanity metrics").

- **Good metrics have the following characteristics:**

- They're actionable. They inform those who operate or revise the service when they need to change its behaviour.

- They're comparative. They compare the performance of something over time, or between groups of devices whose members are in different location or have different firmware or hardware versions.

- They're understandable and relevant in an operational context. This means that in addition to raw values like totals, they can provide information like ratios and rates.

- They provide information at the right resolution. You can choose how often you sample, how often you report, and how you average, bin, and graph your metrics. These values all need to be chosen in the domain context of the service you're trying to deliver. For example, providing 1-second reporting on an IoT device's SD card capacity generates a lot of unnecessary detail and volume. And looking only at CPU load averaged per hour will absorb and hide short, service-crushing spikes in activity. There might be periods of troubleshooting where you dial up the fidelity of metrics for better diagnostics. But the baseline resolution should be appropriate for what you need in order to meet your monitoring needs.

- They illuminate the difference between symptoms, causes, and correlations across what you're measuring. Some measurements are leading indicators of a problem, and you might want to build alerting on those. Other measurements are lagging indicators and help you understand what has happened; these measurements are often used for exploratory analysis.

TEXT / REFERENCE BOOKS

1 Jan Holler, VlasiosTsiatsis, Catherine Mulligan, Stefan Avesand, StamatisKarnouskos, David Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence Press, 2014.

2 Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.

3 Hassan, Qusay F., ed. Internet of Things A to Z: technologies and applications. John Wiley & Sons, 2018.

4 Holler, Jan, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand, and David Boyle. Internet of Things. Academic Press, 2014.

5 Hersent, Olivier, David Boswarthick, and Omar Elloumi. The internet of things: Key applications and protocols. John Wiley & Sons, 2011.

6 Bernd Scholz- -3-642-19156-5 e-ISBN Architecting the Internet of Things, ISBN 978-3-642-19156-5, 978-3-642-19157-2, Springer

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – III– IoT Architecture and its Protocols– SCSA1408

UNIT 3 IoT DATA LINK LAYER & NETWORK LAYER PROTOCOLS

PHY/MAC Layer (3GPP MTC, IEEE 802.11, IEEE 802.15), Wireless HART, ZWave, Bluetooth Low Energy, Zigbee Smart Energy, DASH7 - Network Layer-IPv4, IPv6, 6LoWPAN, 6TiSCH, ND, DHCP, ICMP, RPL, CORPL, CARP wireless LAN technology based on IEEE 802.11 standard.

- **LAN (and WLAN)** continues to be important technology for M2M and IoT applications. This is due to the high bandwidth, reliability, and legacy of the technologies. Where power is not a limiting factor, and high bandwidth is required, devices may connect seamlessly to the Internet via Ethernet (IEEE 802.3) or Wi-Fi (IEEE 802.11).

- The IEEE 802.11 (Wi-Fi) standards continue to evolve in various directions to improve certain operational characteristics depending on usage scenario.

- A widely adopted recent release was IEEE 802.11n, which was specifically designed to enhance throughput (typically useful for streaming multimedia).

- Ongoing work such as IEEE 802.11ac is developing an even higher throughputversion to replace this, focusing efforts in the 5 GHz band.

- IEEE 802.11ah is working on an evolution of the 2007 standard that will allow a number of networked devices to cooperate in the ,1 GHz (ISM) band.

- The idea is to exploit collaboration (relaying, or networking in other words) to extend range, and improve energy efficiency (by cycling the active periods of the radio transceiver).

- The standard aims to facilitate the rapid development of IoT and M2M applications that could exploit burst-like transmissions, such as in metering applications.

- **Bluetooth Low Energy** (BLE; "Bluetooth Smart") is a recent (2010) integration of Nokia's Wibree (2006) standard with the main Bluetooth standard (originally developed and maintained as IEEE 802.15.1 and Bluetooth SIG).

- It is designed for short-range (50 m) applications in healthcare, fitness, security, etc., where high data rates (millions of bits per second) are required to enable application functionality.

- It is deliberately low cost and energy efficient by design, and has been integrated into the majority of recent smartphones.

- With **IPv6 Networking**, attention is paid to the ongoing work to facilitate the use of IP to enable interoperability irrespective of the physical and link layers (i.e. making the fact that devices are networked, with or without wires, with various capabilities in terms of range and bandwidth, essentially seamless).

- **6LoWPAN** (IPv6 Over Low Power Wireless Personal Area Networks) was developed initially by the 6LoWPAN Working Group (WG) of the IETF as a mechanism to transport IPv6 over IEEE 802.15.4-2003 networks.

- **RPL** (IPv6 Routing Protocol for Low Power and Lossy Networks) was developed by the IETF Routing over Low Power and Lossy Networks (RoLL) WG. They defined Low Power Lossy Networks as those typically characterized by high data loss rates, low data rates, and general instability.

- **CoAP** (Constrained Application Protocol) is being developed by the IETF Constrained RESTful Environments (CoRE) WG as a specialized web transfer

protocol for use with severe computational and communication constraints typically characteristic of M2M and IoT applications.

LTE—Long Term Evolution

Long Term Evolution enhanced Machine-Type Communication (LTE eMTC) standards-based technologies support CAT-0 and CAT-M modes. While LPWAN LTE CAT-0 is commonly used to implement M2M/IoT, CAT-M reduces complexity keeping the coverage aspect using existing mobile cellular network infrastructure . LTE eMTC counts on the same mobile technology benefits as security, privacy, data reliability and device identification.

With the applications involving the communication information generated by the human-to-human (H2H) services, the transmission data rate in the cellular networks increased considerably in the last decade. However, the traffic generated by M2M communication has different characteristics from those generated by H2H services in actual mobile technologies. M2M devices send more traffic than receive traffic when compared to H2H devices. For example, (H2H) service traffic has specific concentration characteristics at certain times of the day, while M2M traffic may present a uniform characteristic. In applications, such as measuring, M2M traffic tends to be periodic and mobility is short when compared to (H2H) communication devices. Thus, service quality requirements between M2M and H2M may be very different.

3GPP offered the appropriateness of LTE to permit MAC connection and PHY links through LTE networks and to optimize the technologies and mechanisms related to radio access. It provides LTE networks updates do deal with MTC and IoT requirements as follows:

- CAT-0, version 12 (R12): Since Category 1 (CAT-1) has low transmission capacity, a new category has become necessary to support the new challenges of MTC and IoT. The Category 0, or CAT-0 of R12 have lower hardware complexity when compared to CAT-1
- CAT-M, version 13 (R13): With the objective of new complexity reduction techniques, CAT-M was proposed in R13
- CAT-N, version 13 (R13): As major IoT devices, or MTCs, typically deal with long distance coverage to transmit few data bytes, CAT-N has been incorporated into the LTE specifications to support the required functionality. The main objectives of CAT-N is to improve distance coverage reducing its complexity and, as a consequence, a greater battery life cycle

In order to fulfill the prerequisites necessary for LTE networks to be able to attend MTC services, 3GPP worked on the R12 to minimize power consumption and cost with new data traffic profile, hardware simplification and spectrum adjustments. LTE OFDMA modulation and SC-FDMA coding is maintained. 3GPP LTE Release 13 is being studied for IoT applications due to the fact that it uses a lower transmission bandwidth than the LTE terminals of previous releases. Release 13 works with freedom of spectrum occupancy within the LTE carrier using only 1.4 MHz of the 20 MHz available LTE carriers. A 15 dB link budget enhancement can guarantee longer distance coverage and better obstacle penetration factor. LTE eMTC technology presents the same layers observed in the LTE protocol stack, shown in Figure. Briefly, the layers can be described as follows:

- Non-Access Stratum (NAS): Works between Device and Network Core and is used for control, authentication and mobility management purposes.

- Radio Resource Control (RRC): In an eNodeB base station, this layer takes handover decisions, sends broadcast messages containing system information and controls the measurements of the device (UE) parameters.
- Packet Data Control Protocol (PDCP): Responsible for the compression and decompression of the user IP packet headers. It also performs data encryption, both on the user data plane and on the control plane data plane.
- Radio Link Control (RLC): Used to format and transport data between the Device and eNodeB base station, transfer upper layer Protocol Data Units (PDUs), error correction, concatenation, segmentation, and reassembly of Service Data Unit (SDUs).
- MAC: It performs physical transport channel mapping.
- Physical Layer: This layer carries all the information from the transport channels through the air interface.

With LTE-RACH fixed on the 1.08 MHz bandwidth, it is convenient to use LTE-eMTC channelization in multiples of its value, allowing compatibility with previous versions. The sub-frame structure of the LTE eMTC is the same as the LTE standard. From a frequency plane perspective, the 180 kHz bandwidth of a resource block is divided into 12 sub-carriers, each of them with 15 kHz bandwidth.

In terms of coverage, the LTE eMTC's have a 155 dB link budget is achieved with the addition of 15 dB in channel prediction and resource tuning processes, enabling coverage of up to 11 km. These characteristics allow the LTE eMTC to have satisfactory communication even in conditions of excessive losses but transmission rates are affected [107]. In the downlink, the sub-frame structure of the LTE eMTC uses only a part of the LTE framework legacy. As LTE eMTC devices can be implemented with a narrow bandwidth, it is not possible to use the LTE control channels. Thus, it is necessary to create a MTC Physical Downlink Control Channel (MTC-PDSCH) and Physical Downlink Shared Channel (PDSCH) using LTE data channels. PDSCH transports higher data layers, segmented according to the transmission resources available.

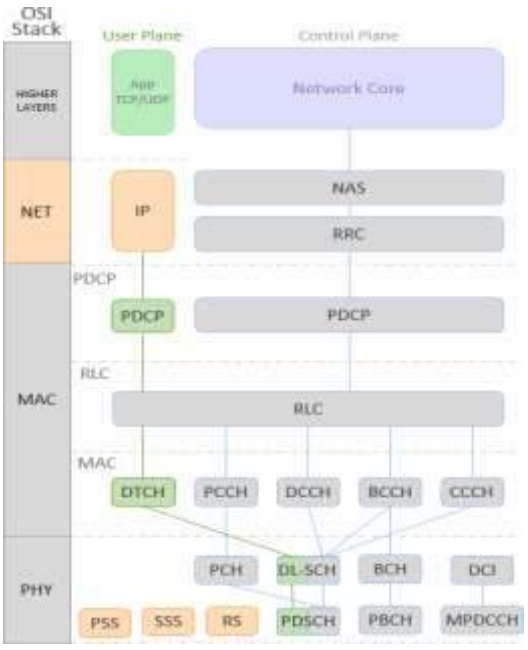


Figure . 3.1 LTE-eMTC downlink layers.

The LTE eMTC devices depend on the NPSS and NSSS of the LTE standard for the acquisition of the carrier frequency of a cell and frame time usage. LTE signals can be used unchanged by LTE eMTC devices even under challenging coverage conditions. To improve reception performance, LTE eMTC devices can accumulate data received in buffer. This happens because the NPSS and NSSS signals are transmitted periodically. In order to discover the communication channel characteristics, the NRS signal is used to give estimates of channel behavior. This signal, which has a known pattern, is transmitted by the base station and performs downlink quality measurements.

For uplink data transmission, there is only one Physical Uplink Control Channel (PUCCH) that carries information such as scheduling requests, channel quality information, and transmission confirmations. The channel used by the user equipment to transmit data in uplink is the Physical Uplink Shared Channel (PUSH). Some sub-carriers can be used to assign resources to the user and are not used by the PUCCH or Physical Random Access Channel (PRACH) signals. To initiate the connection between the terminal and the base station and estimate the arrival time of the uplink message, the PRACH channel is used. Figure illustrates the LTE eMTC uplink channels.

Power consumption on IoT devices consists of standby and active power consumption (Idle and Connected modes, respectively). The power consumption of standby mode depends on the design and technology used, and essentially should not differ between LTE MTC and NB-IoT (both technologies have a battery life of approximately 10 years). Active energy consumption differs between these two technologies. Operating on active power consumption for downlink transmission, LTE eMTC has substantially higher throughput (both bandwidth and higher order modulation) than NB-IoT. As a result, it is possible to obtain an estimated active energy consumption 50% lower than the NB-IoT.

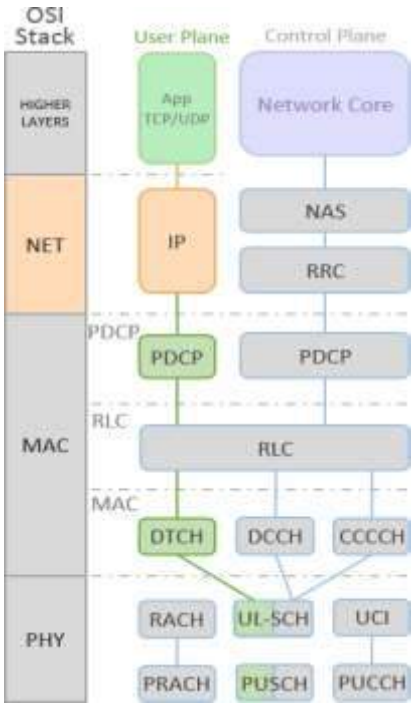


Figure 3.2 LTE eMTC uplink channels.

IEEE 802.11 Architecture

Each computer, mobile, portable or fixed, is referred to as a *station* in 802.11. The difference between a portable and mobile station is that a portable station moves from point to point but is onlyused at a fixed point. Mobile stations access the LAN during movement. Fundamental to the IEEE 802.11 architecture is the concept of Basic Service Set (BSS) or wireless LAN cell. A BSS is defined as a group of stations that coordinate their access to the medium under a given instance of medium access control. The geographic area covered by a BSS is known as the *Basic Service Area (BSA),* which is very similar to a cell in a cellular communication network. All stations with in a BSA with tens of meters in diameter may communicate with each other directly. The 802.11 standard support the formation of two distinct types of BSSs: ad hoc network and Infrastructure BSS.

Two or more BSS's are interconnected using a *Distribution System or DS*. This concept of DS increases network coverage. Each BSS becomes a component of an extended, larger network. Entryto the DS is accomplished with the use of *Access Points (AP).* An access point is a station, thus addressable. So data moves between the BSS and the DS with the help of these access points.

Creating large and complex networks using BSS's and DS's leads us to the next level of hierarchy, the *Extended Service Set or ESS*. The beauty of the ESS is the entire network looks like anindependent basic service set to the Logical Link Control layer (LLC). This means that stations within the ESS can communicate or even move between BSS's transparently to the LLC.

The first type of BSS is known as *ad hoc network,* which consists of a group of stations withinthe range of each other. As its name implies, ad hoc networks are temporary in nature, which are typically created and maintained as needed without prior administrative arrangement. Ad hoc networks can be formed anywhere spontaneously and can be disbanded after a limited period of time. A typical ad hoc network is shown in Figure below.

The second type of BSS is known as *infrastructure BSS (IBSS),* which is commonly used in practice. Here, several BSSs are interconnected by a distribution system to form an extended service set

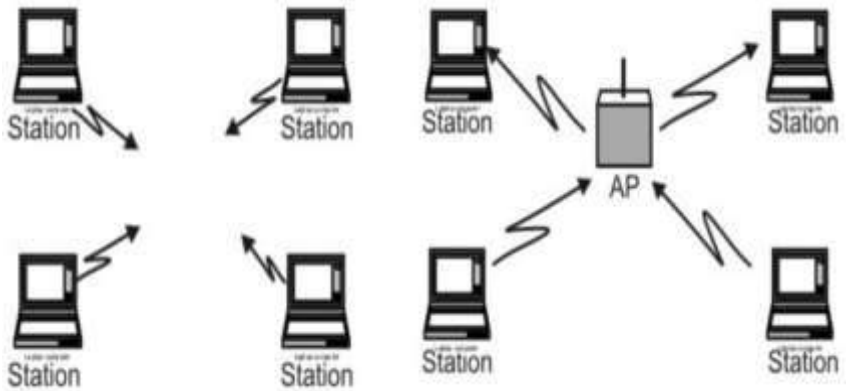


Figure3.3 (a) Basic Service set (BSS), (b) Infrastructure BSS (ESS)

(ESS) as shown in Fig. (b). The BSSs are like cells in a cellular communications network. Each BSS is provided with an Access point (AP) that has station functionality and provides access to the distribution system. APs operate on a fixed channel and remain stationary like *base stations* in a cellular communication system. APs are located such that the BSSs they serve overlap slightly to provide continuous service to all the stations.

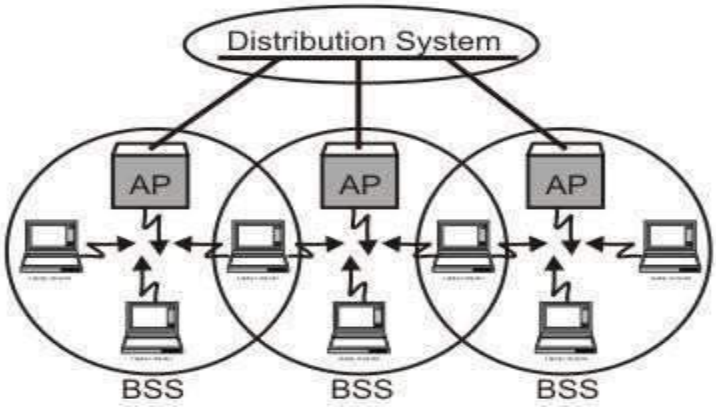


Figure 3.4 Extended service set (ESS)

An ESS can also provide gateway access for wireless users into a wired network. Each end station associates itself with one access point. Above Figure shows three BSSs interconnected throughthree APs to a distribution system. If station A associated with AP-1 wants to send a frame to anotherstation associated with AP- 2, the first sends a frame to its access point (AP-1), which forwards the frame across the distribution system to the access point AP-2. AP-2 finally delivers it to the destination station.

The technique used for this purpose is known as *scanning*, which involves the following steps:

- **A station sends a *probe frame*.**
- **All APs within reach reply with a *probe response frame*.**
- **The station selects one of the access points, and sends the AP an *Association Request frame*.**
- **The AP replies with an *Association Response frame*.**

The above protocol is used when a station joins a network or when it wants to discontinue association with the existing AP because of weakened signal strength or some other reason. The discontinuation of association takes place whenever a station acquires a new AP and the new AP announces it in step 4 mentioned above. For example, assume that station B is moving away from the BSS of AP-1 towards the BSS of AP-2. As it moves closer to the BSS of AP-2, it sends probe frames, which is responded eventually by AP-2. As some of point of time station B prefers AP-2 over AP-1 and associates itself with the access point AP-2. The above mechanism is known as *active scanning*, as the node is actively searching for an access point. An access point also periodically sends Beacon frame that advertises the capabilities of the

access point. In response, a station can associate to the AP simply by sending it an Association request frame. This is known as *passive scanning*.

Medium Access Control:

Most wired LANs products use Carrier Sense Multiple Access with Collision Detection (CSMA/CD) as the MAC protocol. Carrier Sense means that the station will listen before it transmits. If there is already someone transmitting, then the station waits and tries again later. If no one is transmitting then the station goes ahead and sends what it has. But when more than one station tries to transmit, the transmissions will collide and the information will be lost. This is where Collision Detection comes into play. The station will listen to ensure that its transmission made it to the destination without collisions. If a collision occurred then the stations wait and try again later. The time the station waits is determined by the back off algorithm. This technique works great for wired LANs but wireless topologies can create a problem for CSMA/CD. However, the wireless medium presents some unique challenges not present in wired LANs that must be dealt with by the MAC used for IEEE 802.11. Some of the challenges are:

- **The wireless LAN is prone to more interference and is less reliable.**
- **The wireless LAN is susceptible to unwanted interception leading to security problems.**
- **There are so called *hidden station* and *exposed station* problems.**

In the discussion of both the problem, we shall assume that all radio transmitters have fixed range. When the receiver is in the range of two active transmitters then the signal will be garbled. It is important to note that not all stations are in range of two transmitters.

The Hidden Station Problem

Consider a situation when A is transmitting to B, as depicted in the Fig. If C senses the media, it will not hear anything because it is out of range, and thus will falsely conclude that no transmission is going on and will start transmit to B. the transmission will interfere at B, wiping out the frame from A.

The problem of a station not been able to detect a potential competitor for the medium because the competitor is too far away is referred as *Hidden Station Problem.* As in the described scenario C act as a hidden station to A, this is also competing for the medium.

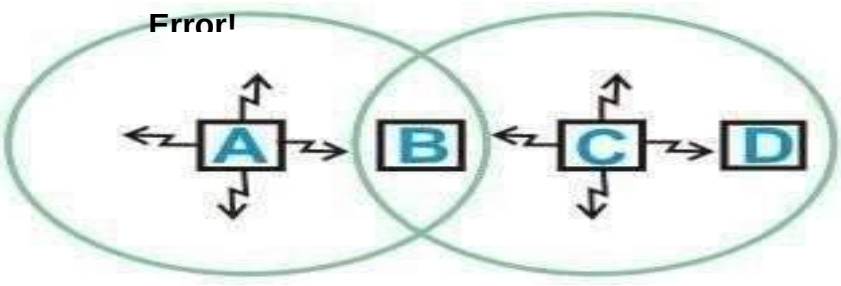


Fig 3.5 Hidden Station Problems

Exposed Station problem

Now consider a different situation where B is transmitting to A, and C sense the medium and detects the ongoing transmission between B and A. C falsely conclude that it can not transmit to D, when the fact is that such transmission would cause on problem. A transmission could cause a problem only when the destination is in zone between B and C. This problem is referred as *Exposedstation Problem.* In this scenario as B is exposed to C, that's why C assumes it cannot transmit to D. So this problem is known as *Exposed station problem* (i.e. problem caused due to exposing of a station). The problem here is that before transmission, a station really wants to know that whether or not there is any activity around the receiver. CSMA merely tells whether or not there is any activity around the station sensing the carrier. Security

Wireless LANs are subjected to possible breaches from unwanted monitoring. To overcome this problem, IEEE 802.11 specifies an optional MAC layer security system known as *Wired Equivalent Privacy* (WEP). The objective is to provide a level of privacy to the wireless LAN similarto that enjoyed by wired Ethernets. It is achieved with the help of a 40-bit shared key authentication service. By default each BSS supports up to four 40-bit keys that are shared by all the clients in the BSS. Keys unique to a pair of communicating clients and direction of transmission may also be used. Advanced Encryption Standard (AES) (802.11i) for authentication and encryption is recommended as a long-term solution.

Frame Control Field (in MAC header)

- **The protocol version field is 2 bits in length and will carry the version of the**

802.11 standard. The initial value of 802.11 is 0; all other bit values are reserved.

- Type and subtype fields are 2 and 4 bits, respectively. They work togetherhierarchically to determine the function of the frame.
- The remaining 8 fields are all 1 bit in length.
- The To DS field is set to 1 if the frame is destined for the distribution system.
- From DS field is set to 1 when frames exit the distribution system. Note that frames whichstay within their basic service set have both of these fields set to 0.
- The More Frag field is set to 1 if there is a following fragment of the current MSDU.
- Retry is set to 1 if this frame is a retransmission.
- Power Management field indicates if a station is in power save mode (set to 1) or active (set to 0).
- More data field is set to 1 if there is any MSDUs are buffered for that station.

- The WEP field is set to 1 if the information in the frame body was processed with theWEP algorithm.
- The Order field is set to 1 if the frames must be strictly ordered.
- The Duration/ID field is 2 bytes long. It contains the data on the duration value for each fieldand for control frames it carries the associated identity of the transmitting station.
- The address fields identify the basic service set, the destination address, the source address,and the receiver and transmitter addresses. Each address field is 6 bytes long.
- The sequence control field is 2 bytes and is split into 2 subfields, fragment number andsequence number.
- Fragment number is 4 bits and tells how many fragments the MSDU is broken into.
- The sequence number field is 12 bits that indicates the sequence number of the MSDU. Theframe body is a variable length field from 0 - 2312. This is the payload.

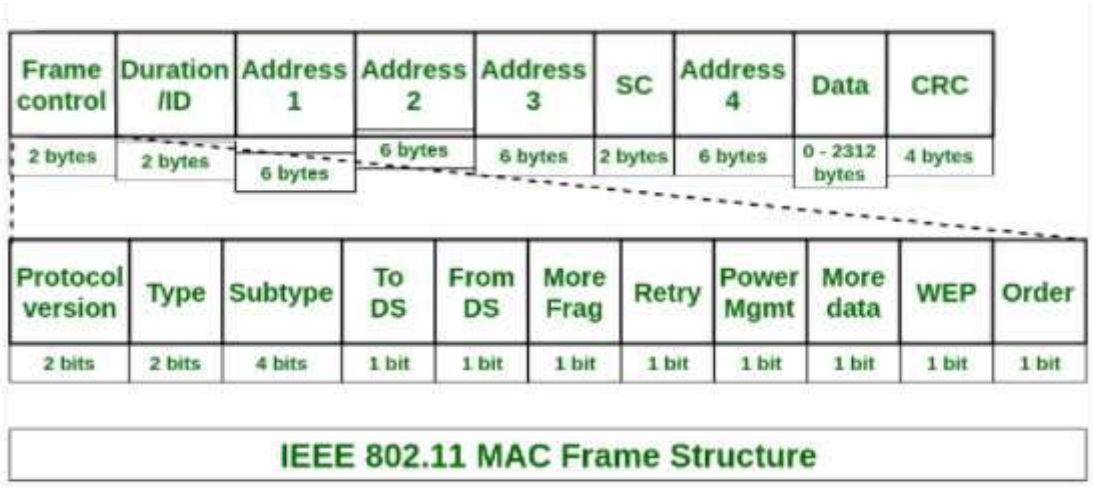
IEEE 802.11 Mac Frame

MAC layer provides functionality for several tasks like control medium access, can also offer support for roaming, authentication, and power conservation. The basic services provided by MAC are the mandatory asynchronous data service and optional time-bounded service.
IEEE 802.11 defines two MAC sub-layers:-

1. **Distributed Coordination Function (DCF) –**
   DCF uses CSMA/CA as access method as wireless LAN can't implement CSMA/CD. It only offers asynchronous service.
2. **Point Coordination Function (PCF) –**
   PCP is implemented on top of DCF and mostly used for time-service transmission. It uses a centralized, contention-free polling access method. It offers both asynchronous and time-bounded service.

MAC Frame:

The MAC layer frame consists of 9 fields. The following figure shows the basic structure of an IEEE 802.11 MAC data frame along with the content of the frame control field.



- **Frame Control(FC) –**
  It is 2 bytes long field which defines type of frame and some control information. Various fields present in FC are:

1. **Version:**
   It is a 2 bit long field which indicates the current protocol version which is fixed to be 0 for now.
2. **Type:**
   It is a 2 bit long field which determines the function of frame i.e management(00), control(01) or data(10). The value 11 is reserved.
3. **Subtype:**
   It is a 4 bit long field which indicates sub-type of the frame like 0000 for association request, 1000 for beacon.
4. **To DS:**
   It is a 1 bit long field which when set indicates that destination frame is for DS(distribution system).
5. **From DS:**
   It is a 1 bit long field which when set indicates frame coming from DS.
6. **More frag (More fragments):**
   It is 1 bit long field which when set to 1 means frame is followed by other fragments.
7. **Retry:**
   It is 1-bit long field, if the current frame is a retransmission of an earlier frame, this bit is set to 1.
8. **Power Mgmt (Power management):**
   It is 1-bit long field that indicates the mode of a station after successful transmission of a frame. Set to 1 the field indicates that the station goes into power-save mode. If the field is set to 0, the station stays active.
9. **More data:**
   It is 1-bit long field that is used to indicate receiver that a sender has more data to send than the current frame. This can be used by an access point to indicate to a station in power-save mode that more packets are buffered or it can be used by a station to indicate to an access point after being polled that more polling is necessary as the station has more data ready to transmit.
10. **WEP:**
    It is 1 bit long field which indicates that the standard security mechanism of 802.11 is applied.
11. **Order:**
    It is 1 bit long field, if this bit is set to 1 the received frames must be processed in strict order.

- **Duration/ID –**
  It is 4 bytes long field which contains the value indicating the period of time in which the medium is occupied(in μs).

- **Address 1 to 4 –**
  These are 6 bytes long fields which contain standard IEEE 802 MAC addresses (48 bit each). The meaning of each address depends on the DS bits in the frame control field.

- **SC (Sequence control) –**
  It is 16 bits long field which consists of 2 sub-fields, i.e., Sequence number (12 bits) and Fragment number (4 bits). Since acknowledgement mechanism frames may be duplicated hence, a sequence number is used to filter duplicate frames.

- **Data –**
  It is a variable length field which contain information specific to individual frames which is transferred transparently from a sender to the receiver(s).

- **CRC (Cyclic redundancy check) –**
  It is 4 bytes long field which contains a 32 bit CRC error detection sequence to ensure error free frame.

Bluetooth Low Energy

Bluetooth low energy or Bluetooth smart is a short range communication protocol with PHY and MAC layer widely used for in-vehicle networking. Its low energy can reach ten times less than the classic Bluetooth while its latency can reach 15 times. Its access control uses a contention-less MAC with low latency and fast transmission. It follows master/slave architecture and offers two types of frames: adverting and data frames. The Advertising frame is used for discovery and is sent by slaves on one or more of dedicated advertisement channels. Master nodes sense advertisement channels to find slaves and connect them. After connection, the master tells the slave it's waking cycle and scheduling sequence. Nodes are usually awake only when they are communicating and they go to sleep otherwise to save their power.

Being part of the Bluetooth v4.0 standard adopted in 2010-06-30, Bluetooth Low Energy (BLE) is also known as Smart Bluetooth. BLE is an IEEE 802.15.1 variation with better and more suitable capacities for low power applications than the classic Bluetooth Basic Rate. Devices that demand communication with both standards of Bluetooth are required to implement and support both protocol stacks due the incompatibilities among them. Star is the only topology accepted by BLE due the standard definition that does not permit physical link connections among slave devices. Any data exchanged between two slave devices shall pass through the unique master and a slave device may not be connected to two master units at the same time. These premises define the formation of a BLE star pico-net .

Using a similar protocol stack as classic Bluetooth, the differences between them starts above the L2CAP layer. Above the L2CAP layer, BLE is the application layer that uses a set of functionalities, which are not present in the classic Bluetooth specifications. These functionalities are the Attribute Protocol (ATT), the Generic Attribute Profile (GATT), the Security Manager Protocol (SMP) and the Generic Access Profile (GAP). Figure 7 depicts the BLE protocol stack.

The two main roles of BLE are: controller and host. BLE differs from the classical Bluetooth in the controller stack that defines the association methods of the devices. A slave can belong to only one pico-net during an association lifetime, and is synchronized with only one master element.

A Host Controller Interface (HCI) is a communication standard applied between the slave and controller. In the Bluetooth Basic Rate, 79 channels are used with a 1 MHz bandwidth to reduce interference with adjacent channels. In Bluetooth Low Energy, the channels are defined in the 2.400–2.4835 GHz band with a 2 MHz guard band. To achieve scalability, the master device controls the number of hosts associated with it by adjusting the value of the connection interval (*ConnInterval* parameter) between hosts and controllers.

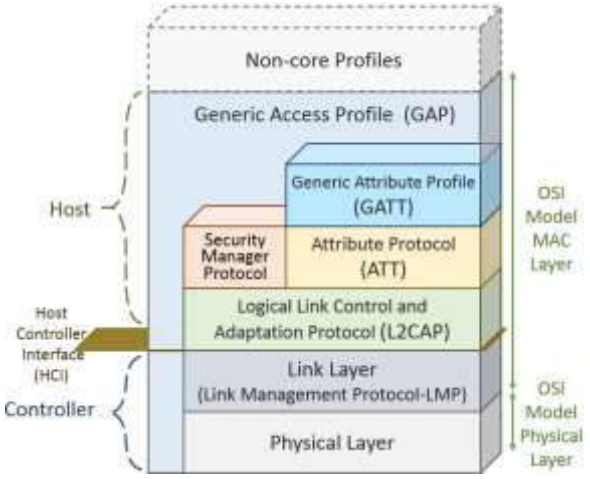


Figure. 3.6 Bluetooth low energy protocol stack.

Link layer manages events generated by the hosts, at determined time intervals, using the advertising channels. Bidirectional data flow is obtained with a connection between elements, when slaves advertising packets are received by master elements. The energy save handling done at MAC layer can put the slaves in a sleeping mode by default and waking them periodically through a Time Division Multiple Access (TDMA) scheme. In the classic Bluetooth basic protocol, this layer a stop-and-wait flow control mechanism is used to provide error recovery capabilities. At BLE, the L2CAP is an adaption of the classic Bluetooth basic protocol stack but optimized and simplified to receive the application layers designed for low energy platforms. Data exchange between the application layer and link layer are also done by L2CAP using no retransmission techniques or flow control mechanisms as used on the classic Bluetooth. Not using retransmission or flow control mechanisms (present in the classic Bluetooth) and segmentation and reassembly capabilities, the Packet Data Units (PDU) (limited to 23 bytes in BLE) received by the application layer is delivered ready to fit the maximum size of the L2CAP payload.

When two devices are connected under a server and client association architecture, the server needs to maintain a set of attributes. The Attribute Protocol (ATT) handles the attributes of this connection like the definition of data structure used to store the information managed by the Generic Attribute Profile (GATT) that works on top of the ATT. GATT defines the client or server functionalities of a connection and this association is independent of the master or slave roles. The attributes of the server need to be accessed by the client through the requests sent, which trigger the response messages of the server. It is also possible for a server to send to a client, unsolicited messages like notifications that do not need any confirmation message to be sent by the client. A server is also required to send indication messages, which need confirmation messages to be sent by the client. The slave sends requests for responses and indications prior to transactions confirmation following a stop-and-wait scheme. Slaves can either write attributes values at the master.

A framework defined by GATT performs the role of discovery services using the ATT attributes, and allows exchange of characteristics between devices interconnected. An attribute carries a set of characteristics that includes a value and properties of the parameter monitored by the device. For example, a humidity sensor needs humidity characteristics and attributes to describe this sensor, and to store its measurements. Thus, this sensor needs a further attribute to specify the measurement units.

Creating specific profiles with the Low Energy Bluetooth standard takes place in the Generic Attribute

Profile (GATT). GATT uses the Attribute Protocol (ATT) protocol in addition to the lower stack protocols, in order to introduce the subdivision of retained server attributes into services and features. Services can contain a set of features, which can include a single value (accessible from the client) and other numerical data that describe such features. Among the assignments of GAP profile specifications are: device role rights, discovery devices and services, as well as establishing connections and security. A new profile based on the existing profile requirements can be created following a profile hierarchy. The interoperability of different devices can be handled through application profiles. Bluetooth is designed to offer a low-cost alternative to Wi-Fi at the expense of the transmission range. Its transmission range is considerably shorter (up to 100 m LOS) and data rate does not exceed 721.2 Kbps in the classic Bluetooth Basic Rate version and can reach 3 Mbps with the Enhanced Data Rate feature. BLE operates at 1Mbps rate on its physical layer, while its application layer can handle only 236.7 Kbps.

In Bluetooth Low Energy, there are no subdivisions in power classes but only the maximum and minimum output power values of the transmitter are provided. Only an approximate value of the maximum reachable distance can be predicted. The low power required for transmission is the main feature of the Bluetooth Low Energy standard and this result is due to enhancements made on the classic version. These enhancements include reduced frequency band and shorter PDU packets. An energy evaluation is offered at using CC2640 radio chipset consumption reference measurements. The comparison is made when operating on 0 dBm transmission power by gathering the main characteristics of Bluetooth and BLE.

Bluetooth v5.0 has no functional block included in its first and second layers when compared to versions v4.0, v4.1, and v4.2. A representation of the inter-layer communication structure and the relationship with Bluetooth layers of different Bluetooth versions can be seen in Figure 8. Device-to-device file transfers, wireless speakers, wireless headsets, and Body Sensor Networks are often enabled with Bluetooth versions.

Finally, some characteristics of the Bluetooth BR and BLE technologies are summarized in Figure 9. This figure allows the comparison of their differences according to the PHY and MAC layer characteristics.

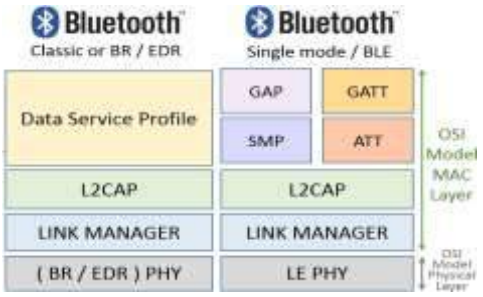


Figure . 3.7 Bluetooth power class classification.

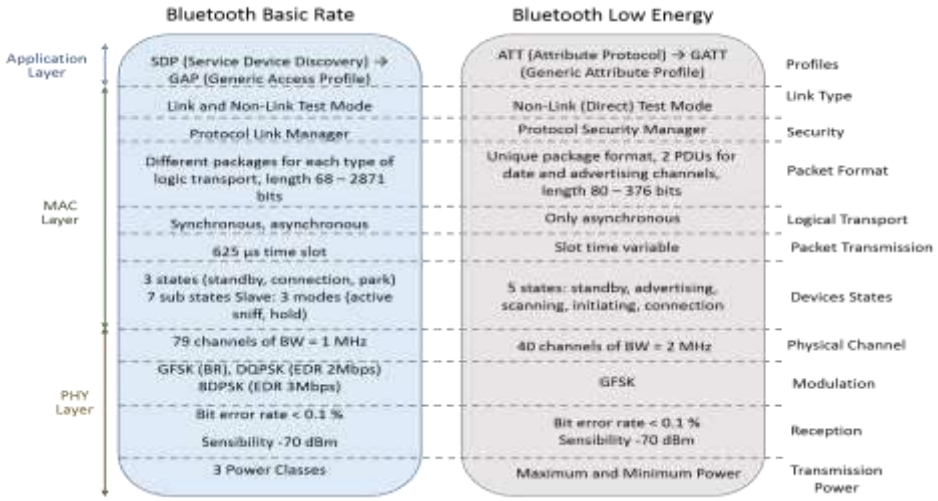


Figure 9. Bluetooth basic rate versus Bluetooth low energy.

IEEE 802.15.4

IEEE 802.15.4 is a subgroup of features that refers to physical and medium access control layers that can support ZigBee and 6LoWPAN upper. IEEE 802.15.4 focuses on physical and data link layer specifications while ZigBee Alliance aims to provide the upper characteristics [33]. It is a standard that defines PHY and MAC layers for personal area networks that demand low rate and low cost applications. This also called a LR-WPAN protocol and has some advantages. Among them are a simple and flexible protocol stack, low cost, low energy consumption, short-range operation, reliable data transfer, and ease of operation [34]. These features are more important when operating in the Personal Operating Space (POS) also defined as Personal Area Network (PAN) that involves the human body.

Physical Layer

The 802.15.4 standard supports an extensive number of PHY options that range from

2.4 GHz to sub-GHz frequencies in ISM bands. (ISM bands are discussed earlier in this chapter.) The original IEEE 802.15.4-2003 standard specified only three PHY options based on direct sequence spread spectrum (DSSS) modulation. DSSS is a modulation technique in which a signal is intentionally spread in the frequency domain, resulting in greater bandwidth. The original physical layer transmission options were as follows:

■2.4 GHz, 16 channels, with a data rate of 250 kbps

■915 MHz, 10 channels, with a data rate of 40 kbps

■868 MHz, 1 channel, with a data rate of 20 kbps

You should note that only the 2.4 GHz band operates worldwide. The 915 MHz band operates mainly in North and South America, and the 868 MHz frequencies are used in Europe, the Middle East, and Africa. IEEE 802.15.4-2006, 802.15.4-2011, and

IEEE 802.15.4-2015 introduced additional PHY communication options, including the following:

■**OQPSK PHY:** This is DSSS PHY, employing offset quadrature phase-shift keying (OQPSK) modulation. OQPSK is a modulation technique that uses four unique bit values that are signaled by phase changes. An offset function that is present during phase shifts allows data to be transmitted more reliably.

■**BPSK PHY:** This is DSSS PHY, employing binary phase-shift keying (BPSK) modulation. BPSK specifies two unique phase shifts as its data encoding scheme.

■**ASK PHY:** This is parallel sequence spread spectrum (PSSS) PHY, employing amplitude shift keying (ASK) and BPSK modulation. PSSS is an advanced encoding scheme that offers increased range, throughput, data rates, and signal integrity compared to DSSS. ASK uses amplitude shifts instead of phase shifts to signal different bit values.

These improvements increase the maximum data rate for both 868 MHz and 915 MHz to 100 kbps and 250 kbps, respectively. The 868 MHz support was enhanced to 3 chan- nels, while other IEEE 802.15.4 study groups produced addendums for new frequency bands. For example, the IEEE 802.15.4c study group created the bands 314–316 MHz, 430–434 MHz, and 779–787 MHz for use in China.

Figure shows the frame for the 802.15.4 physical layer. The synchronization header for this frame is composed of the Preamble and the Start of Frame Delimiter fields. The Preamble field is a 32-bit 4-byte (for parallel construction) pattern that identifies the start of the frame and is used to synchronize the data transmission. The Start of Frame Delimiter field informs the receiver that frame contents start immediately after this byte.

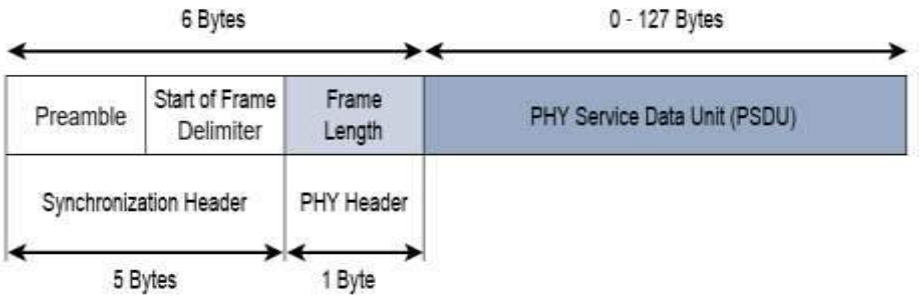


Fig 3.8 PHY header

The PHY Header portion of the PHY frame shown in Figure 4-5 is simply a frame length value. It lets the receiver know how much total data to expect in the PHY service data unit (PSDU) portion of the 802.4.15 PHY. The PSDU is the data field or payload.

MAC Layer

The IEEE 802.15.4 MAC layer manages access to the PHY channel by defining how devices in the same area will share the frequencies allocated. At this layer, the scheduling and routing of data frames are also coordinated. The 802.15.4 MAC layer performs the following tasks:

■Network beaconing for devices acting as coordinators (New devices use beacons to join an 802.15.4 network)

■PAN association and disassociation by a device

■Device security

■Reliable link communications between two peer MAC entities

The MAC layer achieves these tasks by using various predefined frame types. In fact, four types of MAC frames are specified in 802.15.4:

■Data frame: Handles all transfers of data

■Beacon frame: Used in the transmission of beacons from a PAN coordinator

■Acknowledgement frame: Confirms the successful reception of a frame

■flAC command frame: Responsible for control communication between devices

Each of these four 802.15.4 MAC frame types follows the frame format shown in Figure . In Figure, notice that the MAC frame is carried as the PHY payload. The 802.15.4 MAC frame can be broken down into the MAC Header, MAC Payload, and MAC Footer fields.
The MAC Header field is composed of the Frame Control, Sequence Number and the Addressing fields. The Frame Control field defines attributes such as frame type, address- ing modes, and other control flags. The Sequence Number field indicates the sequence identifier for the frame. The Addressing field specifies the Source and Destination PAN Identifier fields as well as the Source and Destination Address fields.
The MAC Payload field varies by individual frame type. For example, beacon frames have specific fields and payloads related to beacons, while MAC command frames have different fields present. The MAC Footer field is nothing more than a frame check sequence (FCS). An FCS is a calculation based on the data in the frame that is used by the receiving side to confirm the integrity of the data in the frame. IEEE 802.15.4 requires all devices to support a unique 64-bit extended MAC address, based on EUI-64. However, because the maximum payload is 127 bytes, 802.15.4 also defines how a 16-bit "short address" is assigned to devices. This short address is local to the PAN and substantially reduces the frame overhead compared to a 64-bit extended MAC address. However, you should be aware that the use of this short address might be limited to specific upper-layer protocol stacks.

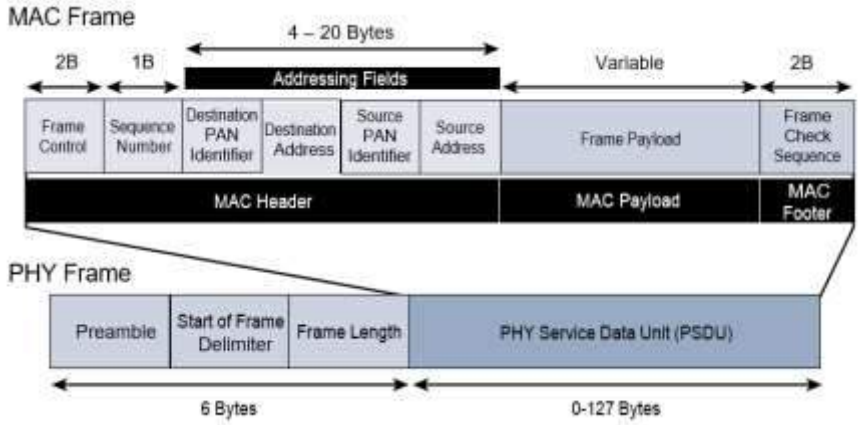


Fig: 3.9 *IEEE 802.15.4 MAC Format*

The IEEE 802.15.4 BE and NBE operational modes have being strongly investigated over recentyears. Thus, some limitations have been addressed and the most important ones are the unbounded delay, low communication efficiency, low interference robustness, and/or fading and main powered relay nodes
Figure compares IEEE 802.15.4 stack with the OSI reference model.

Figure 3.10 IEEE 802.15.4 compared with OSI reference model.

WirelessHART

WirelessHART is a datalink protocol that operates on the top of IEEE 802.15.4 PHY and adopts Time Division Multiple Access (TDMA) in its MAC. It is a secure and reliable MAC protocol that uses advanced encryption to encrypt the messages and calculate the integrity in order to offer reliability. The architecture, as shown in fig consists of a network manager, a security manager, a gateway to connect the wireless network to the wired networks, wireless devices as field devices, access points, routers and adapters. The standard offers end-to-end, per-hop or peer-to- peer security mechanisms. End to end security mechanisms enforce security from sources to destinations while per-hop mechanisms secure it to next hop only.

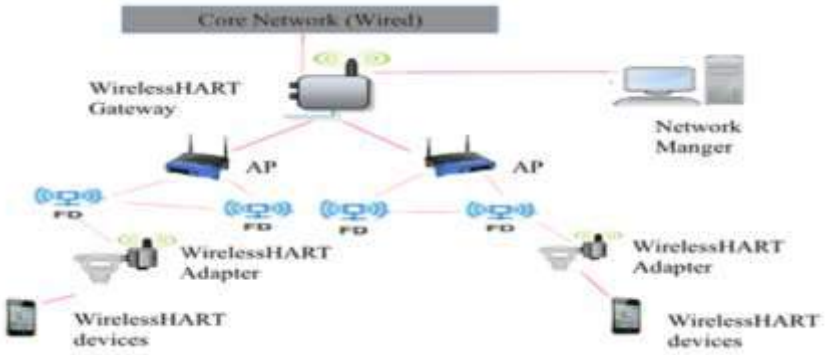


Fig : 3.11 Wireless HART architecture

Wireless-HART (Highway Addressable Remote Transducer Protocol) is a variation of IEEE 802.15.4 design to work essentially as a centralized wireless network. IEEE 802.15.4 is designed to meet the requirements of industrial wireless applications with hard timing parameter restrictions, critically security issues, and severity on obstacle interferences. The Wireless-HART protocol has the same specifications as IEEE 802.15.4 PHY, but develops its own MAC layer based on the TDMA technique. Using Bluetooth, there is no guarantee to delay values on an end-to-end wireless communication.

The absence of a hopping channel technique and a quasi-static star Bluetooth topology works against its scalability. These characteristics make them inappropriate to be used in industrial scenarios. Wireless HART comes as a solution for process control applications through the effort of some industrial organizations such as International Society of Automation 100 (ISA 100) , HART, Wireless Industrial Networking Alliance (WINA) and ZigBee Allianceto attend their specific requirements ratified by the HART Communication Foundation in 2007.

Using the IEEE 802.15.4 PHY layer, Wireless-HART operates in the license-free ISM of 2.4–2.4835 GHz with 2 MHz bandwidth of each one of the 16 channels. The channels are numbered from 11 to 26 with a gap of 5 MHz between IEEE 802.11b/g adjacent channels, delivering up to 250 Kbps. Wireless-HART uses its own Time Division Multiplex Access (TDMA) on the MAC layer including the 10 ms synchronized time slot features. These characteristics allow the messages routing through a network topology obstacle and interference. This is possible due to the use of self-organizing and self-healing mesh networking techniques supported by the network layer. Even being essentially a centralized wireless network, Wireless-HART uses a network manager in its stack in order to provide routing and communication schedules. This can guarantee network performance and satisfy the wireless industrial applications. The focus of Wireless-HART is communication on a one-hop level and the network layer has its responsibility to the network devices vicinity allocation .

Differing from IEEE 802.15.4, Wireless-HART uses time-synchronized the TDMA technique combined with frequency hopping on its MAC layer, thus allowing multiple devices to transmit data at the same time along different channels. During the joining process of the devices onto networks, the network manager distributes the communication links and the channel hop patterns to the devices. It also manages the enabling or disabling of the use of channels that are frequently affected by considerable interference levels, calling this feature *channels blacklist,wHart-n-802-15-4e,petersen2011wirelesshart*. The eight types of devices defined on Wireless-HART are: routers, gateways, adapters, network managers, network security devices, access points and field devices on a mesh topology. All of them support the implementation of features to attend network creation, maintenance issues, data and signaling routing capability, and a minimum of reliability. A comparison between the OSI reference model and Wireless-HART protocol layers and its main features is shown in Figure 12.

Figure. 3.12 WirelessHART Protocol Stack.

Another addressable characteristic of Wireless-HART is the information blocks that each network device maintains on its memory. The information of neighbor nodes and the next reachable device is called a neighbor information block. The connection with the network layer is made through the block information, adding data to the network layer routing table. Working with TDMA as a medium access technique, the network devices have very stringent timing requirements to accomplish network synchronization premises. This happens because synchronization occurs both in the joining process and in normal operations .

Z-Wave

Z-Wave is a low-power MAC protocol designed for home automation and has been used for IoT communication, especially for smart home and small commercial domains. It covers about 30-meter point-to-point communication and is suitable for small messages in IoT applications, like light control, energy control, wearable healthcare control and others. It uses CSMA/CA for collision detection and ACK messages for reliable transmission. It follows a master/slave architecture in which the master control the slaves, send them commands, and handling scheduling of the whole network. Z-Wave was developed and is overseen by the company Zensys to provide wireless communication between devices with a focus on residential automation. Monitoring and controlling of lighting, ambient temperature and security through sensors and actuators by tablets, smartphones or computers are some applications in its portfolio. Z-Wave devices are arranged in mesh network topology. They can send and receive messages from any device that is connected to the network.

The protocol is a proprietary standard based on the ITU G.9959 specification that operates in the Industrial, Scientific, and Medical (ISM) radio frequency band. Z-Wave transmits on 868.42 MHz (Europe) and 908.42 MHz (United States) frequencies working with FSK and Gaussian Fase Shift Keying (GFSK) modulations. With low transmission rates of 9.6 Kbps, 40 Kbps and 100 Kbps, it employs symmetric AES-128 encryption. The MAC layer uses the CSMA-CA technique for a medium access control technique and, based on ITU G.9959, has the following characteristics: a capacity of 232 unique

network identifiers that allows the same quantity of nodes joying the network; collision avoidance mechanism; back-off time when collision occurs; reliability guaranteed by receiving acknowledgments; frame validation and retransmission mechanisms. A power saving mechanism is achieved due to a sleep mode with a dedicated wake-up pattern [61,62]. Figure 13 depicts the Z-Wave protocol stack.

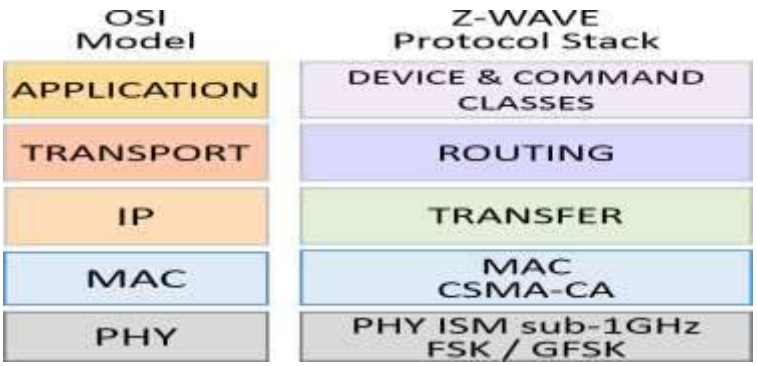


Figure.3.13 Z-Wave protocol stack.

The Z-Wave basic device classes are the following: Portable Controller, Static Controller, Slave, and Slave with Routing Capabilities. Different classes provide the device with a certain role in the Z-Wave network. Inside a Basic Class, Generic and Specific device classes are used to achieve the wanted functionality in the control network. In the Z-Wave protocol, the unique identification of the devices is used through a 32-bit ID. This ID value cannot be changed as it is written in the device chipset by the device manufacturer. A Z-Wave network has only one primary controller device at a time. Each of the 232 nodes of this network can also be a repeater for forwarding data to its neighbors, mediating a connection. Battery-powered nodes do not enjoy this facility. In an environment with a certain level of device drift or even when a device is removed from the network for some reason, the network topology may change. Changing network topology can lead to problems in packet forwarding and packet routing in the network. To minimize this effect, routing tables should be kept up-to-date, optimized and any new topology detected; Z-Wave supports the discovery and suitability of the new network topology. This is possible by keeping the routing table up-to-date on each device and showing all neighboring devices. When a node changes its position or is removed from the network, a topology failure can start an automatic topology and healing procedure to detect the new topology and define the best routes to update the routing tables. This mechanism is subjected to unauthorized modification of routing table attacks by rouge nodes.

The transfer (or transport) layer management functions are: communication between two neighbor nodes, packet acknowledgment, low power network nodes awake (Beaming), and packet origin authentication. This layer controls the *Beam* frames used to wake-up battery powered Z-Wave devices, as each primary controller device of a cluster can handle up to 232 nodes. All nodes can act as a packet repeater, except those devices that are batteries powered. This is Z-Wave mesh topology formed. Z-Wave data security is based on AES and on the cipher block chaining message authentication code (CBC-MAC). However, standards and rules for command classes, device types and timers are missing. These characteristics are only acquired in the new advanced security framework (S2) determined by the Z-Wave Alliance and developed in conjunction with the cyber security community. For the certification of new products as of 2017, Z-Wave brings devices a higher level of security. The structure of S2 is based on the protection of the devices that is already associated with the network, so they are not hacked while still connected to the network. Once the device has already been associated to the network through its pin-code

or QR (Quick Response) code, there is an exchange of security keys through the Elliptic Curve Diffie-Hellman (ECDH) algorithm.

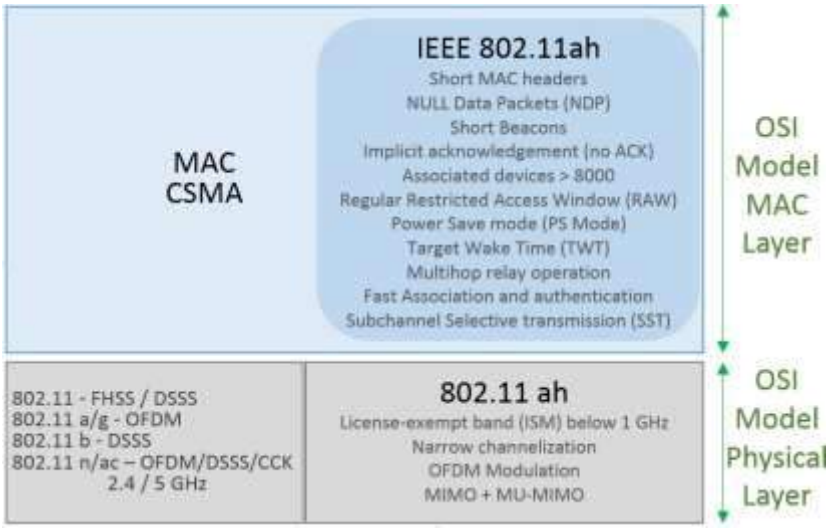


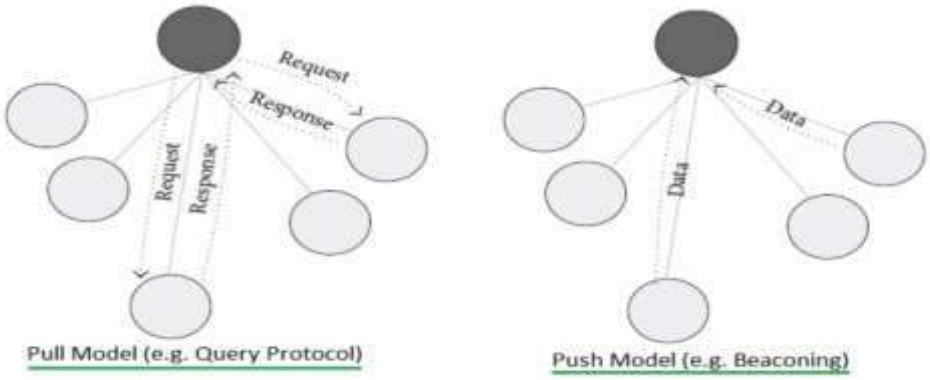
Figure 3.14 IEEE 802.11 standards.

DASH7

DASH7 is a wireless communication protocol for active RFID that operates in globally available Industrial Scientific Medical (ISM) band and is suitable for IoT requirements. It is mainly designed for scalable, long range outdoor coverage with higher data rate compared to traditional ZigBee. It is a low-cost solution that supports encryption and IPv6 addressing. It supports a master/slave architecture and is designed for burst, lightweight, asynchronous and transitive traffic. Its MAC layer features can be summarized as follows

- **Filtering:** Incoming frames are filtered using three processes; cyclic redundancy check (CRC) validation, a 4-bit subnet mask, and link quality assessment. Only the frames that pass all three checks are processed further.
- **Addressing:** DASH7 uses two types of addresses: the unique identifier which is the EUI-64 ID and dynamic network identifier which is 16-bit address specified by the network administrator.
- **Frame format:** The MAC frame has a variable length of maximum 255 bytes including addressing, subnets, estimated power of the transmission and some other optional fields.

The RFID technologies such as NFC and Dash7 is widely used in WSN(wireless sensor networking). Most of the smartphones will have these technologies integrated to provide many facilities to the users. These include building access, mobile payments, advanced location services, home automation, ticketing and more. This means that Dash7 will be part of IoT(Internet of Things) which acquires sensors data and use that to manage social network applications.



Dash7 Communication Models

There are four different device classes defined in D7A (Dash7 Alliance Protocol).

• Blinker:It only transmits and does not use a receiver.

• EndPoint:It can transmit and receive the data. It also supports wake-up events.

• Subcontroller:It is full featured device. It is not always active. It uses wake on scan cycles similar to end points.

• Gateway:It connects D7A network with the other network.It will always be online. It always listens unless it is transmitting.

D7A describes full functional RFID tag. All the devices in Dash7 network supports one or more of the above-mentioned device classes.
DASH7 supports two communication models: pull and push.
The dialogs between tags and interrogators are query response based(referred as pull model). This request response mechanism is described by the D7A Query Protocol Data transfer initiated from the tags to the gateway on the other hand is based on the push model. Both of these models are depicted in the figure. This approach for instance is implemented as an automated message or beacon which is sent on specific time intervals. This system is called Beacon Transmit Series.
DASH7 defines two types of frames viz. a foreground frame and a background frame. The foreground frames are regular messages which contain data or data requests. Background frames on the other hand are very short broadcast messages. Background frames are used by the D7A Advertising Protocol for rapid ad-hoc group synchronization.
Background frame = { subnet(1 byte), payload(3 bytes), CRC16(2 byte) }
Foreground frame = {length(1 byte), headers(3-38 bytes), payload(0 to 249 bytes),
Footer(0 to 20 bytes), CRC16(2 bytes) }

Zigbee Smart Energy

ZigBee Smart Energy (SE) is a standard for interconnecting and interoperating devices, via radio frequency, directed towards monitoring, managing and automating energy, gas and water usage. It seeks to be a useful tool for creating “Green Homes”, and is aimed at coordinating energy usage, optimizing its generation and consumption.

ZigBee SE is a world-leading standard widely used for smart metering (electricity, gas and water) and home automation (wireless domotics). ZigBee is a simple data transmission protocol designed to be used as a low rate wireless personal area network (LR-WPAN). Based on the IEEE 802.15. ZigBee smart energy is designed for a large range of IoT applications including smart homes, remote controls and healthcare systems. It supports a wide range of network topologies including star, peer-to-peer, or cluster-tree. A

coordinator controls the network and is the central node in a star topology, the root in a tree or cluster topology and may be located anywhere in peer-to-peer. ZigBee standard defines two stack profiles: ZigBee and ZigBee Pro. These stack profiles support full mesh networking and work with different applications allowing implementations with low memory and processing power. ZigBee Pro offers more features including security using symmetric-key exchange, scalability using stochastic address assignment, and better performance using efficient many-to-one routing mechanisms.

Wireless protocol for device monitoring and control: Zigbee Smart Energy (Zigbee SE) is a protocol designed for monitoring and actively managing energy consumption at the end-user level. For both utilities and consumers, Zigbee SE can help reduce waste, energy consumption and enables utilities to monitor and manage customers' energy use. Furthermore, the end-user can monitor their energy consumption.

Affordable and easy to use

With Zigbee SE it easier to monitor energy consumption and lower the energy costs – for both utilities and the end-user. Zigbee SE is easy to use because it is interoperable with other Zigbee protocols and works across manufacturers. The standardized protocol provides fast access to new markets for solution providers, and it reduces research costs

Reducing energy consumption

Intelligent energy management with Zigbee SE also benefits the environment, because it improves energy availability. With the access to monitor the energy consumption, both utilities and end-users can regulate the use of energy, thus saving money and reducing the impact on the environment. Increased energy efficiency reduces the need for additional generation plants.

Zigbee Smart Energy features

The Zigbee SE offers real-time information, tracking of energy consumption, multiple control methods e.g. emergency signals and duty cycling, and includes the ability to randomize start and end times to avoid energy spikes. The protocol naturally uses data encryption to safeguard data. The protocol fits multiple commodities, e.g. electricity, gas, and water, and it has several measurement types such as load profile, power factor, demand, and tiers.

ZigBee SE is a world-leading standard widely used for smart metering (electricity, gas and water) and home automation (wireless domotics). ZigBee is a simple data transmission protocol designed to be used as a low rate wireless personal area network (LR-WPAN). Based on the IEEE 802.15.4 specification, for a set of high-level communications protocols, it's a low-powered, low-bandwidth digital radio communication system. Among its most important applications are automation in the home and smart metering. The protocol is intended for devices requiring low-power, relatively low transmission speeds and short distances between devices. In terms of transmission speeds, in the best-case scenario and depending on the frequency used, it can reach 250 Kb/s. The maximum transmission distance can vary from 10 to 100 meters, depending on the frequency, transmission power and environmental conditions. Radio channel access is provided by CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) – a low level network access control protocol that permits multiple stations to use the same transmission medium. ZigBee devices consume very small amounts of energy and are very low cost. They use protocols from different layers (PHY, MAC, network, security, and application). The network, security and application layers are defined by ZigBee Alliance. The networks can work on different frequencies: 868 MHz, 915 MHz and 2.4 GHz. ZigBee networks support around 65,000 devices.

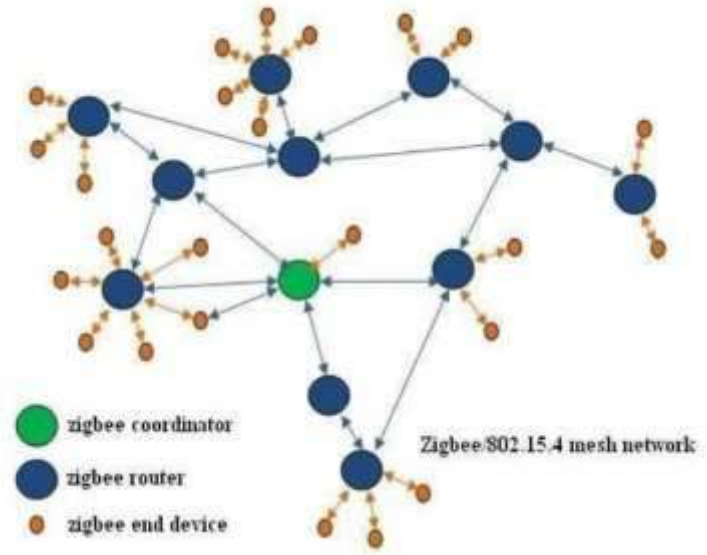
ZigBee networks support star and mesh topologies. Three types of devices are defined:

– **Coordinator:** this is the device that coordinates and forms the network, which means that every network must always have one. Once this device creates the network, other devices (Routers or End devices) can join. It is responsible for selecting the frequency channels and assigning network identifiers (PAN ID) to devices. The PAN ID is used to communicate between network devices. The coordinator can help to route data over mesh networks. It requires a permanent power supply, must always be active and be able to support child devices.

– **Router:** First, it must join the network, after which it can allow other Routers and End devices to join. It requires a permanent power supply, must always be active and be able to support child devices.

– **End Devices**: These do not connect to other network devices. They are usually battery-powered devices and can go into "sleep" mode to save energy.

The following image is a graphical representation of a mesh network structure:



A number of standards use ZigBee as a base. The most common of these are:

ZigBee Home Automation (ZHA): This is a home automation-oriented global standard for controlling applications like lighting, temperature control, energy management, security and accident prevention. **ZigBee Smart Energy (ZSE):** This is a global standard that allows service providers and Home Area Network (HAN) electricity distribution companies to manage energy consumption. ZSE also allows suppliers and customers to interact, so that both can access smart communications. **ZigBee Light Link (ZLL):** This is a global standard that permits consumer lighting elements and other elements to interoperate with each other, giving consumers wireless access to these elements. It allows consumers to control their home lighting, while managing energy use and making their homes "greener".

Reason for ZigBee Smart Energy

ZigBee SE provides service providers and power distributors with a simple wireless access network within homes (Home Area Network, or HAN). Smart Energy offers these groups and their customers the possibility of communicating with each other directly in order to control smart applications (e.g., thermostats and other devices used to control high energy use in the home). Having access to customers'

instantaneous consumption, enables power distributors to more efficiently manage the electricity smart grid (generation and distribution). Furthermore, customers can receive real-time information on their energy use through devices installed inside the home, as well as by accessing the HAN through the services provided by energy distributors and/or service providers.

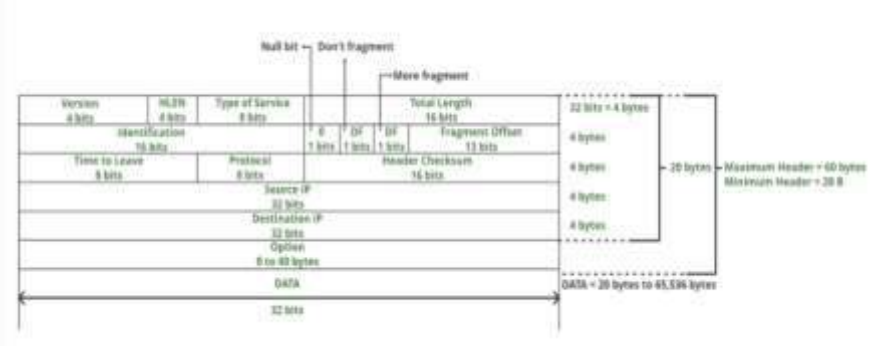
Network Layer Routing Protocols

Internet Protocol Version 4 (IPv4)

IPv4 is 32-bit addressing scheme used as TCP/IP host addressing mechanism. IP addressing enables every host on the TCP/IP network to be uniquely identifiable. IPv4 is a connectionless protocol used for packet-switched networks. It operates on a best effort delivery model, in which neither delivery is guaranteed, nor proper sequencing or avoidance of duplicate delivery is assured. Internet Protocol Version 4 (IPv4) is the fourth revision of the Internet Protocol and a widely used protocol in data communication over different kinds of networks. IPv4 is a connectionless protocol used in packet-switched layer networks, such as Ethernet. It provides a logical connection between network devices by providing identification for each device. There are many ways to configure IPv4 with all kinds of devices – including manual and automatic configurations – depending on the network type. IPv4 is defined and specified in IETF publication RFC 791.IPv4 uses 32-bit addresses for Ethernet communication in five classes: A, B, C, D and E. Classes A, B and C have a different bit length for addressing the network host. Class D addresses are reserved for military purposes, while class E addresses are reserved for future use. IPv4 uses 32-bit (4 byte) addressing, which gives 232 addresses. IPv4 addresses are written in the dot-decimal notation, which comprises of four octets of the address expressed individually in decimal and separated by periods, for instance, 192.168.1.5.

IPv4 Datagram Header

Size of the header is 20 to 60 bytes.



VERSION: Version of the IP protocol (4 bits), which is 4 for IPv4

HLEN: IP header length (4 bits), which is the number of 32 bit words in the header. The minimum value for this field is 5 and the maximum is 15.

Type of service: Low Delay, High Throughput, Reliability (8 bits)

Total Length: Length of header + Data (16 bits), which has a minimum value 20 bytes and the maximum is 65,535 bytes.

Identification: Unique Packet Id for identifying the group of fragments of a single IP datagram (16 bits)

Flags: 3 flags of 1 bit each : reserved bit (must be zero), do not fragment flag, more fragments flag (same order)

Fragment Offset: Represents the number of Data Bytes ahead of the particular fragment in the particular Datagram. Specified in terms of number of 8 bytes, which has the maximum value of 65,528 bytes.

Time to live: Datagram's lifetime (8 bits), It prevents the datagram to loop through the network by restricting the number of Hops taken by a Packet before delivering to the Destination.

Protocol: Name of the protocol to which the data is to be passed (8 bits)

Header Checksum: 16 bits header checksum for checking errors in the datagram header

Source IP address: 32 bits IP address of the sender

Destination IP address: 32 bits IP address of the receiver

Option: Optional information such as source route, record route. Used by the Network administrator to check whether a path is working or not.

Due to the presence of options, the size of the datagram header can be of variable length (20 bytes to 60 bytes).

IPv4 provides hierarchical addressing scheme which enables it to divide the network into sub-networks, each with well-defined number of hosts. IP addresses are divided into many categories:

- **Class A** - it uses first octet for network addresses and last three octets for host addressing
- **Class B** - it uses first two octets for network addresses and last two for host addressing
- **Class C** - it uses first three octets for network addresses and last one for host addressing
- **Class D** - it provides flat IP addressing scheme in contrast to hierarchical structure for above three.
- **Class E** - It is used as experimental.

IPv4 also has well-defined address spaces to be used as private addresses (not routable on internet), and public addresses (provided by ISPs and are routable on internet).

Though IP is not reliable one; it provides 'Best-Effort-Delivery' mechanism.

Internet Protocol Version 6 (IPv6)

IP v6 was developed by Internet Engineering Task Force (IETF) to deal with the problem of IP v4 exhaustion. IP v6 is a 128-bits address having an address space of 2^128, which is way bigger than IPv4. In IPv6 we use Colon-Hexa representation. There are 8 groups and each group represents 2 Bytes.

Exhaustion of IPv4 addresses gave birth to a next generation Internet Protocol version 6. IPv6 addresses its nodes with 128-bit wide address providing plenty of address space for future to be used on entire planet or beyond.

In IPv6 representation, have three addressing methods :

Unicast

Multicast

Anycast

1. Unicast Address –

Unicast Address identifies a single network interface. A packet sent to a unicast address is delivered to the interface identified by that address.

2. Multicast Address –

Multicast Address is used by multiple hosts, called as Group, acquires a multicast destination address. These hosts need not be geographically together. If any packet is sent to this multicast address, it will be distributed to all interfaces corresponding to that multicast address.

3. Anycast Address –

Anycast Address is assigned to a group of interfaces. Any packet sent to an anycast address will be delivered to only one member interface (mostly nearest host possible).

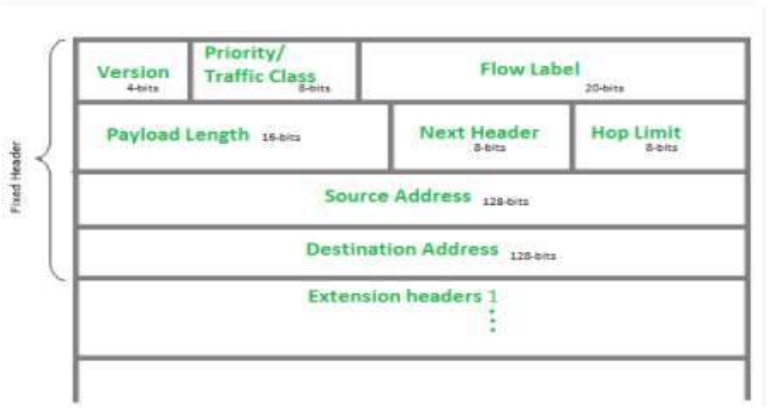
Note: Broadcast is not defined in IPv6.

Types of IPv6 address:

We have 128 bits in IPv6 address but by looking at the first few bits we can identify what type of address it is.

Prefix	Allocation	Fraction of Address Space
0000 0000	Reserved	1/256
0000 0001	Unassigned (UA)	1/256
0000 001	Reserved for NSAP	1/128
0000 01	UA	1/64
0000 1	UA	1/32
0001	UA	1/16
001	Global Unicast	1/8
010	UA	1/8
011	UA	1/8
100	UA	1/8
101	UA	1/8
110	UA	1/8
1110	UA	1/16
1111 0	UA	1/32
1111 10	UA	1/64
1111 110	UA	1/128
1111 1110 0	UA	1/512
1111 1110 10	Link-Local Unicast Addresses	1/1024
1111 1110 11	Site-Local Unicast Addresses	1/1024
1111 1111	Multicast Address	1/256

IP version 6 Header Format :



Version (4-bits): Indicates version of Internet Protocol which contains bit sequence 0110.
Traffic Class (8-bits): The Traffic Class field indicates class or priority of IPv6 packet which is similar to *Service Field* in IPv4 packet. It helps routers to handle the traffic based on the priority of the packet. If congestion occurs on the router then packets with the least priority will be discarded.

As of now, only 4-bits are being used (and the remaining bits are under research), in which 0 to 7 are assigned to Congestion controlled traffic and 8 to 15 are assigned to Uncontrolled traffic.

Priority assignment of Congestion controlled traffic :

Priority	Meaning
0	No Specific traffic
1	Background data
2	Unattended data traffic
3	Reserved
4	Attended bulk data traffic
5	Reserved
6	Interactive traffic
7	Control traffic

Uncontrolled data traffic is mainly used for Audio/Video data. So we give higher priority to Uncontrolled data traffic.The source node is allowed to set the priorities but on the way, routers can change it. Therefore, the destination should not expect the same priority which was set by the source node.

Flow Label (20-bits): Flow Label field is used by a source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers, such as non-default quality of service or real-time service. In order to distinguish the flow, an intermediate router can use the source address, a destination address, and flow label of the packets. Between a source and destination, multiple flows may exist because many processes might be running at the same time. Routers or Host that does not support the functionality of flow label field and for default router handling, flow label field is set to 0. While setting up the flow label, the source is also supposed to specify the lifetime of the flow.

Payload Length (16-bits): It is a 16-bit (unsigned integer) field, indicates the total size of the payload which tells routers about the amount of information a particular packet contains in its payload. The payload Length field includes extension headers(if any) and an upper-layer packet. In case the length of the payload is greater than 65,535 bytes (payload up to 65,535 bytes can be indicated with 16-bits), then the payload length field will be set to 0 and the jumbo payload option is used in the Hop-by-Hop options extension header.

Next Header (8-bits): Next Header indicates the type of extension header(if present) immediately following the IPv6 header. Whereas In some cases it indicates the protocols contained within upper-layer packets, such as TCP, UDP.

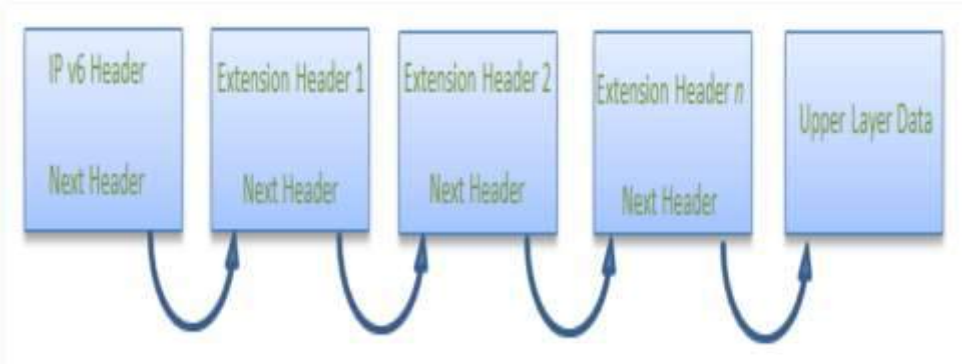
Hop Limit (8-bits): Hop Limit field is the same as TTL in IPv4 packets. It indicates the maximum number of intermediate nodes IPv6 packet is allowed to travel. Its value gets decremented by one, by each node that forwards the packet and the packet is discarded if the value decrements to 0. This is used to discard the packets that are stuck in an infinite loop because of some routing error.

Source Address (128-bits): Source Address is the 128-bit IPv6 address of the original source of the packet.

Destination Address (128-bits): The destination Address field indicates the IPv6 address of the final destination(in most cases). All the intermediate nodes can use this information in order to correctly route the packet.

Extension Headers: In order to rectify the limitations of the IPv4 Option Field, Extension Headers are introduced in IP version 6. The extension header mechanism is a very important part of the IPv6

architecture. The next Header field of IPv6 fixed header points to the first Extension Header and this first extension header points to the second extension header and so on.



IPv6 has introduced Anycast addressing but has removed the concept of broadcasting. IPv6 enables devices to self-acquire an IPv6 address and communicate within that subnet. This auto-configuration removes the dependability of Dynamic Host Configuration Protocol (DHCP) servers. This way, even if the DHCP server on that subnet is down, the hosts can communicate with each other.

IPv6 provides new feature of IPv6 mobility. Mobile IPv6 equipped machines can roam around without the need of changing their IP addresses.

IPv6 is still in transition phase and is expected to replace IPv4 completely in coming years. At present, there are few networks which are running on IPv6. There are some transition mechanisms available for IPv6 enabled networks to speak and roam around different networks easily on IPv4. These are:

- Dual stack implementation
- Tunneling
- NAT-PT

IPv4	IPv6
IPv4 has a 32-bit address length	IPv6 has a 128-bit address length
It Supports Manual and DHCP address configuration	It supports Auto and renumbering address configuration
In IPv4 end to end, connection integrity is Unachievable	In IPv6 end to end, connection integrity is Achievable
It can generate 4.29×10^{9} address space	Address space of IPv6 is quite large it can produce 3.4×10^{38} address space
The Security feature is dependent on application	IPSEC is an inbuilt security feature in the IPv6 protocol
Address representation of IPv4 is in decimal	Address Representation of IPv6 is in hexadecimal
Fragmentation performed by Sender and forwarding routers	In IPv6 fragmentation performed only by the sender
In IPv4 Packet flow identification is not available	In IPv6 packet flow identification are Available and uses the flow label field in the header
In IPv4 checksum field is available	In IPv6 checksum field is not available
It has broadcast Message Transmission Scheme	In IPv6 multicast and anycast message transmission scheme is available
In IPv4 Encryption and Authentication facility not provided	In IPv6 Encryption and Authentication are provided
IPv4 has a header of 20-60 bytes.	IPv6 has header of 40 bytes fixed

Dynamic IP Addresses and DHCP

IP addresses may be static or dynamic. Static means that the address assigned to a machine generally does not change. Dynamic means that an IP address is assigned to a computer on demand, for a fixed lease period. The computer may be assigned a different address each time it demands one. Dynamic addresses are acceptable for a machine running client software, since the way things are organized is that the client initiates communications with a server, and includes its "return address" (the source IP address) in every packet sent to the server. To communicate to a server, it is necessary to find out the numeric IP address of the server before the client can communicate to it. That is often accomplished through the Domain Name System (DNS), essentially tables where the IP address of a server can be looked up. To avoid having to frequently update those tables, servers are generally assigned static addresses.

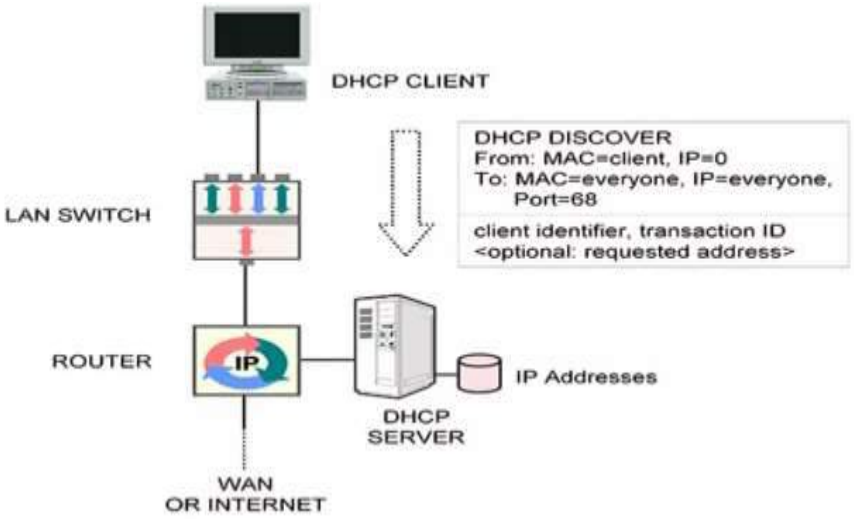


Fig 3.16 DHCP Discover message

Addresses are assigned to a computer using the Dynamic Host Configuration Protocol (DHCP). The system administrator provisions a DHCP server, configured to assign IP addresses within a defined range to clients. Computers are loaded with DHCP client software. Communications between the DHCP client and server are effectively application-layer messages, coded into ASCII and carried in UDP protocol data units, which are carried in IP packets, which are carried in MAC frames. The desired recipient of the messages is indicated as being the DHCP on a machine by populating in the UDP header destination port = 67 for messages to the server and destination port = 68 for messages to the client. The messages are "broadcast", which means that the destination IP address is all 1s and destination MAC address is all 1s. The actual addresses are used for source MAC and IP addresses, except that the client uses "0" as its IP address, since of course the whole point of the exercise is to get an IP address. Each computer will run a DHCP client when it starts, generating a DHCP Discover message as illustrated above.

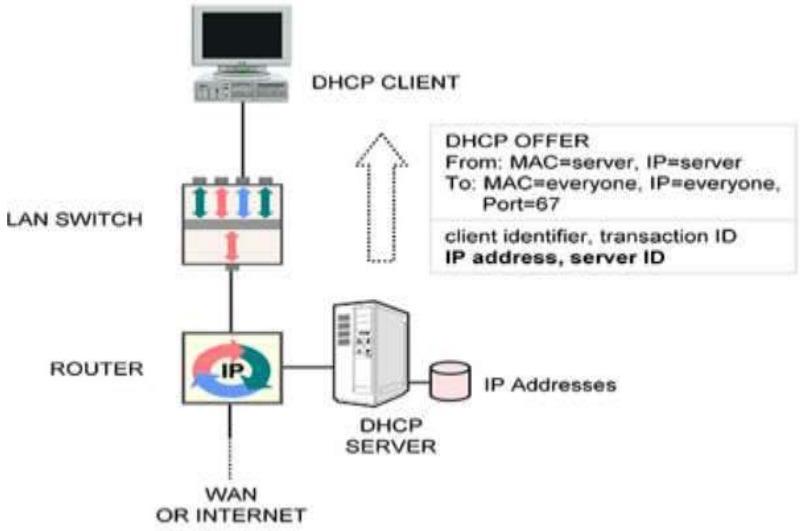


Fig 3.17 DHCP offer message

Any DHCP server that receives it, and there may be more than one, will respond with a DHCP Offer message, with an offered IP address and a lease time:

The client will answer with a DHCP Request message to confirm its selection of an offered address:

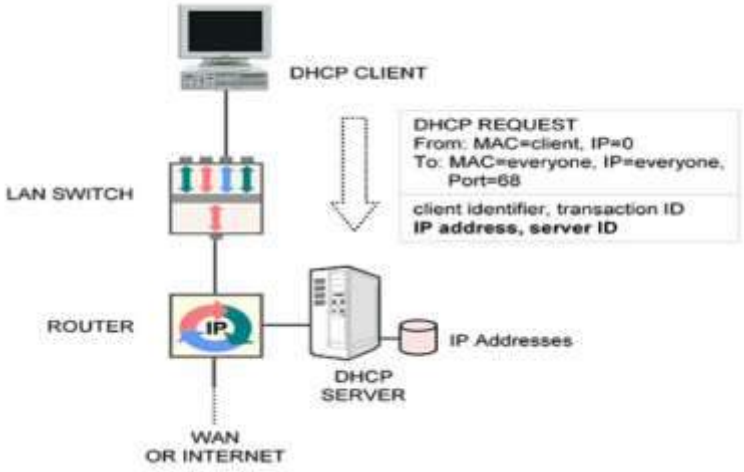


Fig 3.18 DHCP Request message

Then the server will complete the cycle with a DHCP ACK that usually includes other configuration information such as the IP address of the default gateway (CE router), the IP address of one or more DNS servers and the subnet mask, which indicates what bits in the address are the host or machine ID:

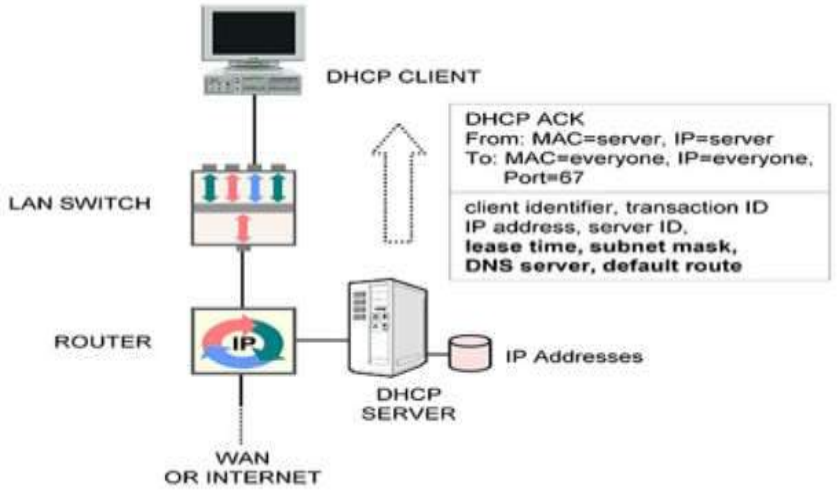


Fig 3.19 DHCP Acknowledge message

Internet Control Message Protocol (ICMP)

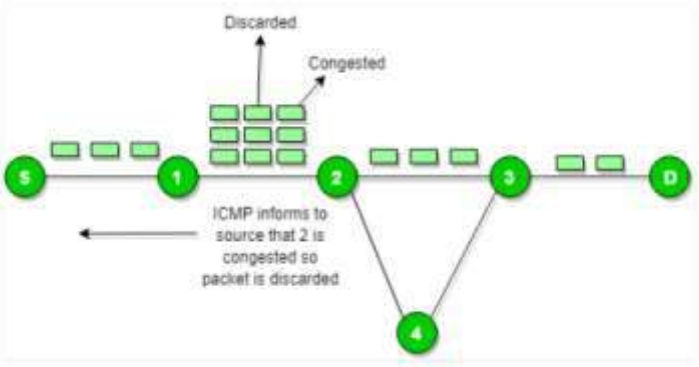
Since IP does not have an inbuilt mechanism for sending error and control messages. It depends on Internet Control Message Protocol(ICMP) to provide an error control. It is used for reporting errors and management queries. It is a supporting protocol and is used by networks devices like routers for sending error messages and operations information., e.g. the requested service is not available or that a host or router could not be reached.

ICMPv4 Packet Format :

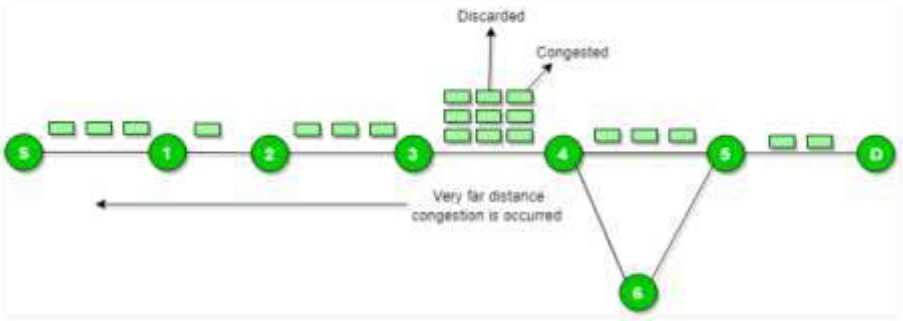
Type(8 bit)	Code(8 bit)	CheckSum(16 bit)
Extended Header(32 bit)		
Data/Payload(Variable Length)		

Source quench message :

Source quench message is a request to decrease the traffic rate for messages sending to the host(destination). Or we can say when receiving host detects that the rate of sending packets (traffic rate) to it is too fast it sends the source quench message to the source to slow the pace down so that no packet can be lost.



ICMP will take the source IP from the discarded packet and informs the source by sending a source quench message. Then source will reduce the speed of transmission so that router will be free from congestion



When the congestion router is far away from the source the ICMP will send hop by hop source quench message so that every router will reduce the speed of transmission.

RPL

Routing Protocol for Low-Power and Lossy Networks (RPL) is distance-vector protocol that can support a variety of datalink protocols.RPL builds a directed acyclic graph (DAG) with no outgoing edges as the base element of the topology, this ensure that no cycles exist in the hierarchy. The sink node starts building the first DAG making itself the ultimate DAG root, other nodes in this DAG start forming their own DAGs which are routed towards the first one making a destination oriented DAG (DODAG). RPL uses a number of control messages to build and maintain its hierarchy. The DODAG information object (DIO) is sent from the root node with information about the rank of the sending node, the instance ID, the version number and the DODAG-ID. This allows nodes to decide whether or not to act upon receiving this message, in addition to keeping valuable information about the network that can contribute to making an informed decision. The destination advertisement object (DAO) is sent from the child node to its parent (the DAG root or the DODAG root) and it contains destination information which practically informs the root that this node is still available. The root node may optionally send a DAOack acknowledgement if required. The DODAG information solicitation is another form of upward control messages that is used to request a DIO from the parent node, this is one of the most relevant and important features that RPL uses to maintain connectivity. Fig 1 shows the direction of RPL control messages.

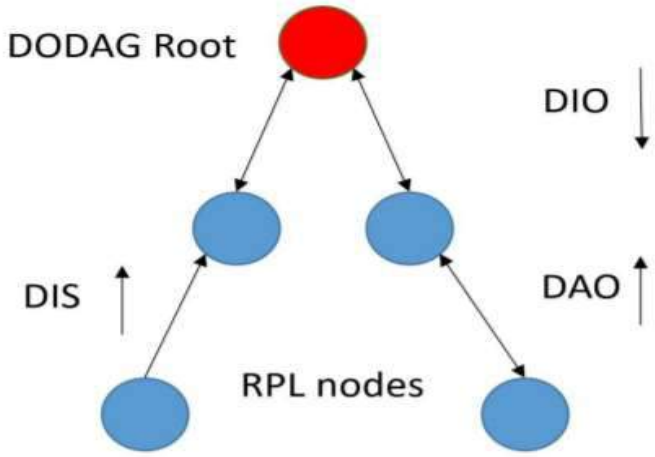


Fig 3.20 Control messages in RPL

CORPL

An extension of RPL is CORPL, or cognitive RPL, which is designed for cognitive networks and uses DODAG topology generation but with two new modifications to RPL. CORPL utilizes opportunistic forwarding to forward the packet by choosing multiple forwarders (forwarder set) and coordinates between the nodes to choose the best next hop to forward the packet to. DODAG is built in the same way as RPL. Each node maintains a forwarding set instead of its parent only and updates its neighbor with its changes using DIO messages. Based on the updated information, each node dynamically updates its neighbor priorities in order to construct the forwarder set.

CARP

Channel-Aware Routing Protocol (CARP) is a distributed routing protocol designed for underwater communication. It can be used for IoT due to its lightweight packets. It considers link quality, which is computed based on historical successful data transmission gathered from neighboring sensors, to select the forwarding nodes. There are two scenarios: network initialization and data forwarding. In network initialization, a HELLO packet is broadcasted from the sink to all other nodes in the networks. In data forwarding, the packet is routed from sensor to sink in a hop- by-hop fashion. Each next hop is determined independently. The main problem with CARP is that it does not support reusability of previously collected data. In other words, if the application requires sensor data only when it changes significantly, then CARP data forwarding is not beneficial to that specific application. An enhancement of CARP was done in E-CARP by allowing the sink node to save previously received sensory data. When new data is needed, E-CARP sends a Ping packet which is replied with the data from the sensors nodes. Thus, E-CARP reduces the communication overhead drastically

Features	RPL	CORPL	CARP
Full Form	Routing Protocol for Low-Power and Lossy Networks	Cognitive RPL	Channel-Aware Routing Protocol
Server technologies	Supported	Supported	Not supported
Security	Not supported	Not supported	Not supported
Storage management	Supported	Not supported	Supported
Data management	Supported	Supported	Supported

Network Layer Encapsulation Protocols

One problem in IoT applications is that IPv6 addresses are too long and cannot fit in most IoT datalink frames which are relatively much smaller. Hence, IETF is developing a set of standards to encapsulate IPv6 datagrams in different datalink layer frames for use in IoT applications. In this section, we review these mechanisms briefly.

6LoWPAN

IPv6 over Low power Wireless Personal Area Network (6LoWPAN) is the first and most commonly used standard in this category. It efficiently encapsulates IPv6 long headers in IEEE802.15.4 small packets, which cannot exceed 128 bytes. The specification supports different length addresses, low bandwidth, different topologies including star or mesh, power consumption, low cost, scalable networks, mobility, unreliability and long sleep time. The standard provides header compression to reduce transmission overhead, fragmentation to meet the 128-byte maximum frame length in IEEE802.15.4, and support of multi-hop delivery. Frames in 6LoWPAN use four types of headers: No 6loWPAN header (00), Dispatch header (01), Mesh header (10) and Fragmentation header (11). In No 6loWPAN header case, any frame that does not follow 6loWPAN specifications is discarded. Dispatch header is used for multicasting and IPv6 header compressions. Mesh headers are used for broadcasting; while Fragmentation headers are used to break long IPv6 header to fit into fragments of maximum 128-byte length.

6TiSCH

6TiSCH working group in IETF is developing standards to allow IPv6 to pass through Time-Slotted Channel Hopping (TSCH) mode of IEEE 802.15.4e datalinks. It defines a Channel Distribution usage matrix consisting of available frequencies in columns and time-slots available for network scheduling operations in rows. This matrix is portioned into chucks where each chunk contains time and frequencies and is globally known to all nodes in the network. The nodes within the same interference domain negotiate their scheduling so that each node gets to transmit in a chunk within its interference domain. Scheduling becomes an optimization problem where time slots are assigned to a group of neighboring nodes sharing the same application. The standard does not specify how the scheduling can be done and

leaves that to be an application specific problem in order to allow for maximum flexibility for different IoT applications. The scheduling can be centralized or distributed depending on application or the topology used in the MAC layer [Dujovne14].

6Lo

IPv6 over Networks of Resource-constrained Nodes (6Lo) working group in IETF is developing a set of standards on transmission of IPv6 frames on various datalinks. Although, 6LowPAN and 6TiSCH, which cover IEEE 802.15.4 and IEEE 802.15.4e, were developed by different working groups, it became clear that there are many more datalinks to be covered and so 6Lo working group was formed. At the time of this writing most of the 6Lo specifications have not been finalized and are in various stages of drafts. For example, IPV6 over Bluetooth Low Energy Mesh Networks, IPv6 over IEEE 485 Master-Slave/Token Passing (MS/TP) networks, IPV6 over DECT/ULE, IPV6 over NFC, IPv6 over IEEE 802.11ah, and IPv6 over Wireless Networks for Industrial Automation Process Automation (WIA-PA) drafts are being developed to specify how to transmit IPv6 datagrams over their respective datalinks [6Lo]. Two of these 6Lo specifications "IPv6 over G.9959" and "IPv6 over Bluetooth Low Energy" have been approved as RFC and are described next.

IPv6 over G.9959

RFC 7428 defines the frame format for transmitting IPv6 packet on ITU-T G.9959 networks. G.9959 defines a unique 32-bit home network identifier that is assigned by the controller and 8-bit host identifier that is allocated for each node. An IPv6 link local address must be constructed by the link layer derived 8-bit host identifier so that it can be compressed in G.9959 frame. Furthermore, the same header compression as in 6lowPAN is used here to fit an IPv6 packet into G.9959 frames. RFC 7428 also provides a level of security by a shared network key that is used for encryption. However, applications with a higher level of security requirements need to handle their end-to-end encryption and authentication using their own higher layer security mechanisms [6Lo].

IPv6 over Bluetooth Low Energy

Bluetooth Low Energy is also known as Bluetooth Smart and was introduced in Bluetooth V4.0 and enhanced in V4.1. RFC 7668 [RFC7668], which specifies IPv6 over Bluetooth LE, reuses most of the 6LowPAN compression techniques. However, since the Logical Link Control and Adaptation Protocol (L2CAP) sublayer in Bluetooth already provides segmentation and reassembly of larger payloads in to 27 byte L2CAP packets, fragmentation features from 6LowPAN standards are not used. Another significant difference is that Bluetooth Low Energy does not currently support formation of multi-hop networks at the link layer. Instead, a central node acts as a router between lower-powered peripheral nodes.

ICMP is actually a user of the IP protocol--in other words, ICMP messages must be encapsulated within IP packets. However, ICMP is implemented as part of the IP layer. So ICMP processing can be viewed as occurring parallel to, or as part of, IP processing. Therefore, in the topic on TCP/IP-based layered network, ICMP is shown as a layer 3 protocol.

ICMP is probably most well known as the message protocol used for the **ping** command. A ping command sends an ICMP echo request to the target host. The target host responds with an echo reply. The ping command is losing some of its usefulness in today's more security-conscious networks: many routers disable responses to echo requests.

ICMP's primary use on a network is to deliver information about simple problems with the delivery of packets. For example, ICMP can inform hosts about:

- Maximum transmission unit limitations. When a packet that is too large for a network to handle arrives at a router, the router will break it into smaller packets (*fragments*).

  If the packet has a flag (an IP flag, in fact) stipulating the packet cannot be fragmented, then the router will discard the packet and send an ICMP *fragmentation needed* packet back to the original sender.

- Packet expiry. The time exceeded after a packet has traversed too many hops.
- Destination unreachable. For example, when an ARP broadcast fails to elicit a matching IP address.
- Routing problems. When a host believes a better route exists.

  Note that this is not a desirable feature of ICMP and should be disabled under almost all circumstances. Routing protocols do a better job of determining the best route.

RPL-Routing Protocol for Low Power and Lossy Networks

RPL is the IPv6 Routing Protocol for LLN (Low-Power and Lossy Networks) as defined in RFC6550. The Routing Protocol for Low Power and Lossy Networks feature implements RPL on Cisco IOS software.

Restrictions for Routing Protocol for Low Power and Lossy Networks

- RPL can be enabled only on the main interface.
- RPL may not work if multiple links have same link local address.
- The probing parameters are not configurable in root template in this implementation. The probing parameters are required in root template also.
- The following RPL features listed in RFC6550 are not supported:
  - Secure RPL Control Messages.
  - Non-storing mode.
  - Multicast DAO
  - Unicast DIS
  - Unicast DIO
  - Reception of DAO-ACK
  - Virtual DODAG root
  - Metrics and constraints—Implements a subset of these metrics and constraints only.
  - DIO Destination prefix
  - Non-storing mode for DAO routes—Supports processing of DAO prefixes received from nonstoring nodes but always operate in a full route storing mode.
  - No support for adding tags to DAO prefixes using DAO suboptions.
  - Local confidence
  - Interface level link check for Expected Transmission Count or latency calculation—Uses layer-3 probe process for calculating link.
- Cannot change the values of the Instance-ID, DODAG ID, and OCP options for an RPL after the RPL template is configured. If you want to change the options, you must delete the template from all interfaces.
- The **no dao enable** command cannot be used when the template is active.

Information About Routing Protocol for Low Power and Lossy Networks

Low Power and Lossy Networks

Low power and Lossy Networks (LLNs) are a class of network in which both the routers and their interconnect are constrained: LLN routers typically operate with constraints on (any subset of) processing power, memory and energy (battery), and their interconnects are characterized by (any subset of) high loss rates, low data rates and instability. LLNs are comprised of anything from a few dozen and up to thousands of LLN routers, and support point-to-point traffic (between devices inside the LLN), point-to-multipoint traffic (from a central control point to a subset of devices inside the LLN) and multipoint-to-point traffic (from devices inside the LLN towards a central control point).

The Routing Protocol for Low Power and Lossy Networks feature specifies the IPv6 Routing Protocol for LLNs (RPL), thereby providing a mechanism whereby multipoint-to-point traffic from devices inside the LLN towards a central control point, and point-to-multipoint traffic from the central control point to the devices inside the LLN, is supported. Point-to-point traffic is also supported.

RPL Routing Attributes (Metrics and Constraints)

RPL makes use of a wide set of routing attributes that can be used either as constraints or metrics as against IGPs such as IS-IS or OSPF. When used as a constraint, the routing attribute allows pruning the links and nodes from candidate paths that do not respect the constraint; when used as a metric the routing attribute is used to determine the least cost path. Additionally, routing metrics can either be used as aggregated (e.g. path cost equal to the sum of the link metrics) or recorded in which case metrics of all links along the path are recorded and announced along with the path by RPL. Recorded metrics are particularly useful when aggregating metrics implies loosing relevant information for path selection.

Metrics can also be local (exchanged between two neighbors) or global (propagated along the path as the path cost).RPL uses dynamic metrics, which implies the use of low pass filters to preserve routing stability, avoid temporary loops and bind control traffic by limiting the number of advertised RPL messages due to link metrics updates and consequently path cost that may have a global impact by recomputing the routing topology.

The Routing Protocol for Low Power and Lossy Networks feature supports aggregated metrics only and the following subset of routing metrics or constraints:

- Routing metrics—ETX (link), Latency (link) and DAG rank.
- Routing constraints—Node State and Attribute (NSA), Node Energy (node).

Expected Transmission Count (ETX)

ETX is defined as the Expected Transmission Count (ETX) metric. ETX is the number of transmissions a node expects to make to a destination in order to successfully deliver a packet. The formula for calculating ETX is

ETX= 1 / (Df * Dr)

where Df is the measured probability that a packet is received by the neighbor and Dr is the measured probability that the acknowledgment packet is successfully received.

ETX can be configured at root or node using the ocp etx command. ETX are exclusively based on IPv6 probing when configured via this command.

The Probing Process

The aim of the probing process is to compute the ETX for all links for which the ETX is required. The probing process is also used to dynamically compute the link latency when latency is used.

Objective Functions

The Objective Function (OF) is used by RPL to specify how the routing metric and constraints should be used to reach specific objectives. For example, the OF may specify that the objective is to find the constrained shortest path where the constraint is related to the node power mode and the metric is the ETX.

Directed Acyclic Graph (DAG) Formation

Each Destination Oriented Directed Acyclic Graph (DODAG) is identified by:

- RPLInstanceID (potentially multiple DODAG, but one OF)
- DODAGID (set by the DAG Root)
- DODAGVersionNumber (DODAG iteration number)
- Parameters advertised by the DODAG root: DAG metric container content, trickle timers, etc.

The DODAGID will be specified as part of the RPL root template definition via the **dag-id** command.

RPL Packet Format

RPL messages are carried via ICMPv6 message. The following messages are available:

- DAG Information Solicitation (DIS)
- DAG Information Object (DIO)
- Destination Advertisement Object (DAO)

RPL DIO Message

RPL DIO messages are sent for DODAG discovery and maintenance. DIO are multicasted according to trickle timers. DIO messages are used for DODAG discovery and maintenance.

Use the **dao solicit-interval** command to configure the frequency (in minutes) at which the root can solicit destination advertisement messages from the downstream nodes. The mode of operation setting is specified as part of instance ID definition. You can also increment the version number using the **version-number** *version* keyword argument pair in the **rpl** command.

Trickle Timers

Control plane traffic load is a concern in LLNs where bandwidth and energy are often scarce. Periodic emission of control plane messages is not possible and the use of keepalive for routing adjacency maintenance is not a suitable option. A different approach in RPL consists of controlling the control plane packets frequency update by using adaptive mechanisms controlled by the use of dynamic timers, referred to as trickle timers. DIO messages are “multicasted” on expiry of trickle timers thereby the DIO messages are sent more frequently when a DAG consistency issue is detected to improve the convergence time. As DAG stabilizes, messages are sent less frequently.

Supported Objective Functions

The Routing Protocol for Low power and Lossy Networks (RPL) feature supports the Objective functions (OF)—OF0, OF1 (ETX) and OF1 (latency). These functions constitute a large set to cover a wide spectrum of uses in Smart Cities. OF0 is used where the objective is to build a DODAG that simply provides connectivity, while OF1 is used to build a reliable network and optimized DODAGs without delay requirements, for example, smart metering and lighting management, and traffic management in a smart city. Each OF specifies a different parent selection criteria and strictly follow the rules for parent selection as specified in the RFC for loop avoidance.

Use the **step-rank** in the RPL template definition to configure the objective function.

Route Redistribution

RPL supports redistribution of routes into a limited set (OSPF, IS-IS, BGP) of routing protocols. The route redistribution happens at the root, which will redistribute routes into other routing protocol that are directly connected as well as routes learnt from DAO messages. Routes learnt from other routing protocols will not be redistributed into the RPL network domain. The root updates the application VRFs with the routing update information. Use the redistribute rpl command to specify redistribution in an RPL network.

RPL Manageability

Management is critical in all networking solution but particularly in environments comprising a large number of networking devices. The DODAG root (a Cisco router) plays a central role in the DODAG management. The Routing Protocol for Low power and Lossy Networks (RPL) feature provides an extensive set of configuration options via CLI (general and service internal) to tune the behavior in the field.

RPL Routing Performance

The unique characteristics of LLNs provide several design choices for RPL. RPL routing performance is dependent on the routing environments. If convergence time is of utmost importance for a routing protocol operating in a “classic” IP networks, the prime performance objective in LLNs is scalability and stability, which are the main challenges in LLNs where links and nodes are potentially highly unstable. Another objective is to bound the control traffic to save bandwidth and energy—the control traffic is negligible compared to the traffic that is considerably lower in “typical” IP networks. The following metrics can be used to evaluate the performances of a routing protocol:

Scalability

RPL is designed to support Home, Building, Smart Grid and Smart Cities networks. Thus, the routing protocol must support from a few dozens to a few thousands of nodes in a single LLN.

RPL IOS has been primarily designed for Smart Grid networks (substation automation, smart metering) and S+CC environments and could also be applied to home and building automation and even in Datacenters.

The objective of RPL is to support a network of 5,000 nodes. In comparison with link state protocols, RPL has been designed to ensure an increase of the number of nodes in the network has a limited impact on the overall scalability of the protocol. The protocol is topology agnostic and does not impact scalability nor the stability (lossiness) of the links.

Control Plane Overhead

RPL uses “trickle” timers whereby the control plane overhead varies with the level of stability of the network. When changes occurs in the networks, routing updates are more frequent with all nodes sending more regular updates while using some level of randomness to avoid collisions (especially critical for low power wireless networks). This allows for reducing the control plane and save bandwidth and energy in the network.

Path Optimization

RPL is a distance vector protocol and supports a wide set of routing link and node metrics. RPL supports mono metric optimization—the best path is considered as the shortest (constrained) path according to a single metric (multimetric optimization is not supported). The objective is to not trade path optimality for network stability. A small path cost increase is usually smoothed out for the benefit of limiting the control plane traffic.

Convergence Time

A number of techniques have been developed over the past few years to improve routing protocol convergence time. Fast recovery techniques relying on either protection or restoration have been designed to achieve convergence between a few seconds and a few hundreds of milliseconds. RPL has been designed to support two repair mechanisms: Global Repair and local repair.

Convergence in RPL networks is controlled by several factors, such as trickle timers, local repair parameters and Neighbor Unreachability Detection (NUD). Depending on the deployment requirements, the DODAG can be built with new values to achieve faster or slower convergence. In the case of NUD, the convergence will depend on the frequency of data traffic in the DODAG.

For deployments where applications use multicast traffic, it may be required to fine-tune RPL parameters (for example trickle-timer configuration) for that specific DODAG to achieve faster convergence to avoid sending redundant multicast traffic to stale node.

Flexibility or Extensibility

RPL supports an object-TLV based approach allowing for the flexible addition of functionalities in future such as new routing metrics, constraints and objective functions that could be defined in the future. Additionally, due to its the “root”-based nature where the DAG root (a Cisco router as opposed to a constraint sensor device), sophisticated functions could be added to the DAG root with limited or negligble impact on more constrained routers in the network.

Local Confidence

The notion of local confidence in RPL is used to determine when a link with another RPL router or leaf is valid and usable to build a router adjacency and also to declare that an existing link used by RPL should no longer be used. In comparison with other routing protocols such decision is rather simple and determined by the link layer. Lossy links have a tendency to flap and transient states should not trigger routing adjacency changes at the risk of routing instability, control traffic churn, etc. Although the notion is fairly generic and not standardized the local confidence is built on a number of inputs.

In RPL, the local confidence is exclusively driven by the use of NUD, IPv6 probing (when enabled) for OF0 augmented by the smoothed calculation of the link ETX with OF1 that does smooth out the transient phenomenon of a low pass filter.

TEXT / REFERENCE BOOKS

1 Jan Holler, VlasiosTsiatsis, Catherine Mulligan, Stefan Avesand, StamatisKarnouskos, David Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence Press, 2014.

2 Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.

3 Hassan, Qusay F., ed. Internet of Things A to Z: technologies and applications. John Wiley & Sons, 2018.

4 Holler, Jan, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand, and David Boyle. Internet of Things. Academic Press, 2014.

5 Hersent, Olivier, David Boswarthick, and Omar Elloumi. The internet of things: Key applications and protocols. John Wiley & Sons, 2011.

6 Bernd Scholz- -3-642-19156-5 e-ISBN Architecting the Internet of Things, ISBN 978-3-642-19156-5, 978-3-642-19157-2, Springer

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – IV– IoT Architecture and its Protocols– SCSA1408

UNIT 4 TRANSPORT & SESSION LAYER PROTOCOLS

Transport Layer: TCP, MPTCP, UDP, DCCP, SCTP-TLS, DTLS Session Layer: HTTP, CoAP, XMPP, AMQP, MQTT

The Transport Layer

With the TCP/IP protocol, two main protocols are specified for the transport layer:

- Transmission Control Protocol (TCP): This connection-oriented protocol requires a session to get established between the source and destination before exchanging data. You can view it as an equivalent to a traditional telephone conversation, in which two phones must be connected and the communication link established before the parties can talk.
- User Datagram Protocol (UDP): With this connectionless protocol, data can be quickly sent between source and destination—but with no guarantee of delivery. This is analogous to the traditional mail delivery system, in which a letter is mailed to a destination. Confirmation of the reception of this letter does not happen until another letter is sent in response.

With the predominance of human interactions over the Internet, TCP is the main protocol used at the transport layer. This is largely due to its inherent characteristics, such as its ability to transport large volumes of data into smaller sets of packets. In addition, it ensures reassembly in a correct sequence, flow control and window adjustment, and retransmission of lost packets. These benefits occur with the cost of overhead per packet and per session, potentially impacting overall packet per second performances and latency. In contrast, UDP is most often used in the context of network services, such as Domain Name System (DNS), Network Time Protocol (NTP), Simple Network Management Protocol (SNMP), and Dynamic Host Control Protocol (DHCP), or for real-time data traffic, including voice and video over IP. In these cases, performance and latency are more important than packet retransmissions because re-sending a lost voice or video packet does not add value. When the reception of packets must be guaranteed error free, the application layer protocol takes care of that function. When considering the choice of a transport layer by a given IoT application layer proto- col, it is recommended to evaluate the impact of this choice on both the lower and upper layers of the stack. For example, most of the industrial application layer protocols, as discussed later in this chapter, are implemented over TCP, while their specifications may offer support for both transport models. The reason for this is that often these industrial application layer protocols are older and were deployed when data link layers were often unreliable and called for error protection. While the use of TCP may not strain generic compute platforms and high-data-rate networks, it can be challenging and is often overkill on constrained IoT devices and networks. This is particularly true when an IoT device needs to send only a few bytes of data per transaction. When using TCP, each packet needs to add a minimum of 20 bytes of TCP overhead, while UDP adds only 8 bytes. TCP also requires the establishment and potential maintenance of an open logical channel. IoT nodes may also be limited by the intrinsic characteristics of the data link layers. For example, low-power and lossy networks (LLNs). A new IoT application protocol, such as Constrained Application Protocol (CoAP), almost always uses UDP and why implementations of indus-trial application layer protocols may call for the optimization and adoption of the UDP transport layer if run over LLNs. For example, the Device Language Message Specification/Companion Specification for Energy Metering (DLMS/COSEM) applica- tion layer protocol, a popular protocol for reading smart meters in the utilities space, is the de facto standard in Europe. Adjustments or optimizations to this protocol should be made depending on the IoT transport protocols that are present in the lower layers. For example, if you compare the transport of DLMS/COSEM over a cellular network versus an

LLN deployment, you should consider the following:

- Select TCP for cellular networks because these networks are typically more robust and can handle the overhead. For LLNs, where both the devices and network itself are usually constrained, UDP is a better choice and often mandatory.
- DLMS/COSEM can reduce the overhead associated with session establishment by offering a “long association” over LLNs. Long association means that sessions stay up once in place because the communications overhead necessary to keep a session established is much less than is involved in opening and closing many sepa- rate sessions over the same time period. Conversely, for cellular networks, a short association better controls the costs by tearing down the open associations after transmitting.
- When transferring large amounts of DLMS/COSEM data, cellular links are preferred to optimize each open association. Smaller amounts of data can be handled effi- ciently over LLNs. Because packet loss ratios are generally higher on LLNs than on cellular networks, keeping the data transmission amounts small over LLNs limits the retransmission of large numbers of bytes.

Multicast requirements are also impacted by the protocol selected for the transport layer. With multicast, a single message can be sent to multiple IoT devices. This is useful in the IoT context for upgrading the firmware of many IoT devices at once. Also, keep in mind that multicast utilizes UDP exclusively.

To guarantee interoperability, certification and compliance profiles, such as Wi-SUN, need to specify the stack from Layer 1 to Layer 4. This enables the chosen technology to be compatible with the different options of the stack while also being compatible with IP.

TRANSPORT LAYER

Transport protocols such as TCP make use of this service to provide applications with reliable, in-order, data stream delivery. They use either UDP or TCP as a transport mechanism. Remember that UDP is unreliable and offers no flow-control, so in this case, the application has to provide its own error recovery, flow control, and congestion control functionality. It is often easier to build applications on top of TCP because it is a reliable stream, connection-oriented, congestion-friendly, flow control-enabled protocol. As a result, most application protocols will use TCP, but there are applications built on UDP to achieve better performance through increased protocol efficiencies. Most applications use the client/server model of interaction

TCP (Transmission Control Protocol):

TCP is a layer 4 protocol which provides acknowledgement of the received packets and is also reliable as it resends the lost packets. It is better than UDP but due to these features it has an additional overhead. It is used by application protocols like HTTP and FTP.

UDP (User Datagram Protocol):

UDP is also a layer 4 protocol but unlike TCP it doesn’t provide acknowledgement of the sent packets. Therefore, it isn’t reliable and depends on the higher layer protocols for the same. But on the other hand it is simple, scalable and comes with lesser overhead as compared to TCP. It is used in video and voice streaming.

TCP Vs UDP –

Session Multiplexing:

A single host with a single IP address is able to communicate with multiple servers. While using TCP, first a connection must be established between the server and the receiver and the connection is closed when the transfer is completed. TCP also maintains reliability while the transfer is taking place.

UDP on the other hand sends no acknowledgement of receiving the packets. Therefore, provides no reliability.

Segmentation:

Information sent is first broken into smaller chunks for transmission.

Maximum Transmission Unit or MTU of a Fast Ethernet is 1500 bytes whereas the theoretical value of TCP is 65495 bytes. Therefore, data has to be broken into smaller chunks before being sent to the lower layers. MSS or Maximum Segment Size should be set small enough to avoid fragmentation. TCP supports MSS and Path MTU discovery with which the sender and the receiver can automatically determine the maximum transmission capability.

UDP doesn't support this; therefore it depends on the higher layer protocols for data segmentation.

Flow Control:

If sender sends data faster than what receiver can process then the receiver will drop the data and then request for a retransmission, leading to wastage of time and resources. TCP provides end-to-end flow control which is realized using a sliding window. The sliding window sends an acknowledgement from receiver's end regarding the data that the receiver can receive at a time.

UDP doesn't implement flow control and depends on the higher layer protocols for the same.

Connection Oriented:

TCP is connection oriented, i.e., it creates a connection for the transmission to take place, and once the transfer is over that connection is terminated. UDP on the other hand is connectionless just like IP (Internet Protocol).

Reliability:

TCP sends an acknowledgement when it receives a packet. It requests a retransmission in case a packet is lost. UDP relies on the higher layer protocols for the same.

Headers: The size of TCP header is 20-bytes (16-bits for source port, 16-bits for the destination port, 32-bits for seq number, 32-bits for ack number, 4-bits header length).The size of the UDP header is 8-bytes (16-bits for source port, 16-bits for destination port, 16-bits for length, 16-bits for checksum); it's significantly smaller than the TCP header. Both UDP and TCP header is comprised of 16-bit Source port(these are used for identifying the port number of the source) fields and 16-bits destination port (these are used for specifying the offered application) fields.

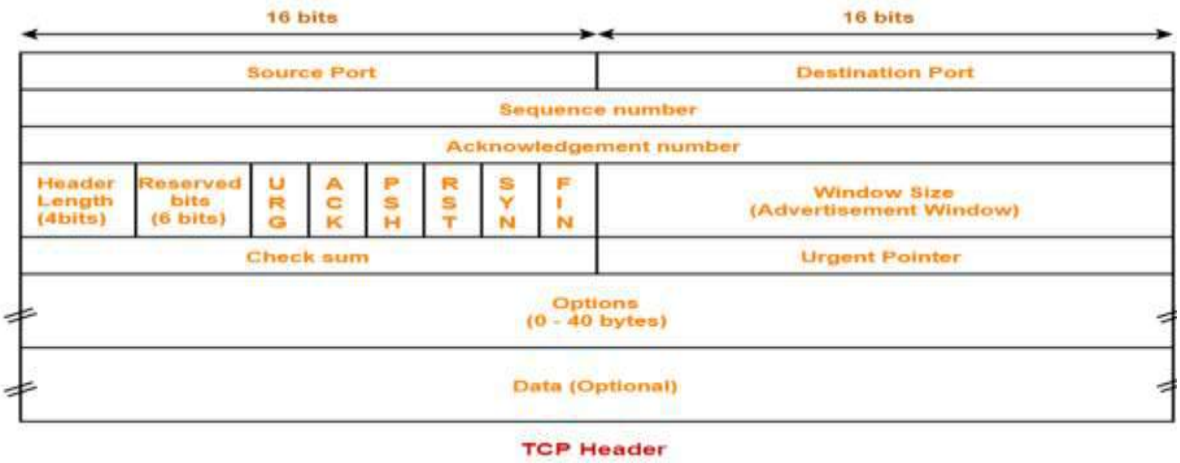


Fig 4.1: TCP header

Source port: this is a 16 bit field that specifies the port number of the sender.

Destination port: this is a 16 bit field that specifies the port number of the receiver.

Sequence number: the sequence number is a 32 bit field that indicates how much data is sent during the TCP session. When you establish a new TCP connection (3 way handshake) then the initial sequence number is a random 32 bit value. The receiver will use this sequence number and sends back an acknowledgment. Protocol analyzers like wireshark will often use a relative sequence number of 0 since it's easier to read than some high random number.

Acknowledgment number: this 32 bit field is used by the receiver to request the next TCP segment. This value will be the sequence number incremented by 1.

DO: this is the 4 bit data offset field, also known as the header length. It indicates the length of the TCP header so that we know where the actual data begins.

RSV: these are 3 bits for the reserved field. They are unused and are always set to 0.

Flags: there are 9 bits for flags, we also call them control bits. We use them to establish connections, send data and terminate connections:

URG: urgent pointer. When this bit is set, the data should be treated as priority over other data.

ACK: used for the acknowledgment.

PSH: this is the push function. This tells an application that the data should be transmitted immediately and that we don't want to wait to fill the entire TCP segment.

RST: this resets the connection, when you receive this you have to terminate the connection right away. This is only used when there are unrecoverable errors and it's not a normal way to finish the TCP connection.

SYN: we use this for the initial three way handshake and it's used to set the initial sequence number.

FIN: this finish bit is used to end the TCP connection. TCP is full duplex so both parties will have to use the FIN bit to end the connection. This is the normal method how we end an connection.

Window: the 16 bit window field specifies how many bytes the receiver is willing to receive. It is used so the receiver can tell the sender that it would like to receive more data than what it is currently receiving. It does

so by specifying the number of bytes beyond the sequence number in the acknowledgment field.

Checksum: 16 bits are used for a checksum to check if the TCP header is OK or not.

Urgent pointer: these 16 bits are used when the URG bit has been set, the urgent pointer is used to indicate where the urgent data ends.

Options: this field is optional and can be anywhere between 0 and 320 bits.

UDP protocol

In computer networking, the UDP stands for User Datagram Protocol. The David P. Reed developed the UDP protocol in 1980. It is defined in RFC 768, and it is a part of the TCP/IP protocol, so it is a standard protocol over the internet. The UDP protocol allows the computer applications to send the messages in the form of datagrams from one machine to another machine over the Internet Protocol (IP) network. The UDP is an alternative communication protocol to the TCP protocol (transmission control protocol). Like TCP, UDP provides a set of rules that governs how the data should be exchanged over the internet. The UDP works by encapsulating the data into the packet and providing its own header information to the packet. Then, this UDP packet is encapsulated to the IP packet and sent off to its destination. Both the TCP and UDP protocols send the data over the internet protocol network, so it is also known as TCP/IP and UDP/IP. There are many differences between these two protocols. UDP enables the process-to-process communication, whereas the TCP provides host to host communication. Since UDP sends the messages in the form of datagrams, it is considered the best-effort mode of communication. TCP sends the individual packets, so it is a reliable transport medium. Another difference is that the TCP is a connection-oriented protocol whereas, the UDP is a connectionless protocol as it does not require any virtual circuit to transfer the data. UDP also provides a different port number to distinguish different user requests and also provides the checksum capability to verify whether the complete data has arrived or not; the IP layer does not provide these two services. The User Datagram Protocol (UDP) is simplest Transport Layer communication protocol available of the TCP/IP protocol suite. It involves minimum amount of communication mechanism. UDP is said to be an unreliable transport protocol but it uses IP services which provides best effort delivery mechanism. In UDP, the receiver does not generate an acknowledgement of packet received and in turn, the sender does not wait for any acknowledgement of packet sent. This shortcoming makes this protocol unreliable as well as easier on processing.

Requirement of UDP

Deploy UDP where the acknowledgement packets share significant amount of bandwidth along with the actual data. For example, in case of video streaming, thousands of packets are forwarded towards its users. Acknowledging all the packets is troublesome and may contain huge amount of bandwidth wastage. The best delivery mechanism of underlying IP protocol ensures best efforts to deliver its packets, but even if some packets in video streaming get lost, the impact is not calamitous and can be ignored easily. Loss of few packets in video and voice traffic sometimes goes unnoticed.

Features

- UDP is used when acknowledgement of data does not hold any significance.
- UDP is good protocol for data flowing in one direction.
- UDP is simple and suitable for query based communications.
- UDP is not connection oriented.

- UDP does not provide congestion control mechanism.
- UDP does not guarantee ordered delivery of data.
- UDP is stateless.
- UDP is suitable protocol for streaming applications such as VoIP, multimedia streaming.

The fields in a UDP header are:

Source port – The port of the device sending the data. This field can be set to zero if the destination computer doesn't need to reply to the sender.

Destination port – The port of the device receiving the data. UDP port numbers can be between 0 and 65,535.

Length – Specifies the number of bytes comprising the UDP header and the UDP payload data. The limit for the UDP length field is determined by the underlying IP protocol used to transmit the data.

Checksum – The checksum allows the receiving device to verify the integrity of the packet header and payload. It is optional in IPv4 but was made mandatory in IPv6.

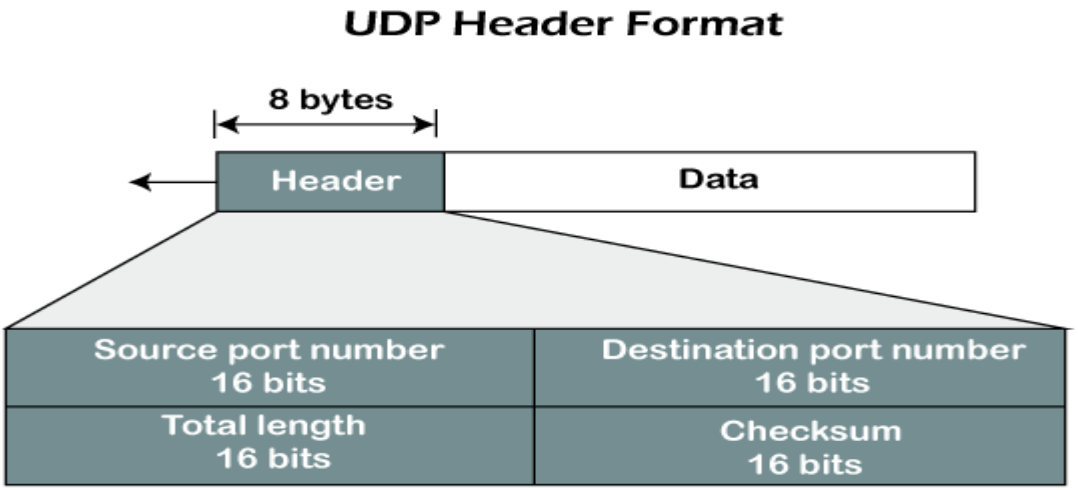


Fig 4.2: UDP header

The following figure shows the encapsulation of a UDP datagram as a single IPv4 datagram.

The IPv6 encapsulation is similar, but other details differ slightly.

The IPv4 Protocol field has the value 17 to indicate UDP.

IPv6 uses the same value (17) in the Next Header field.

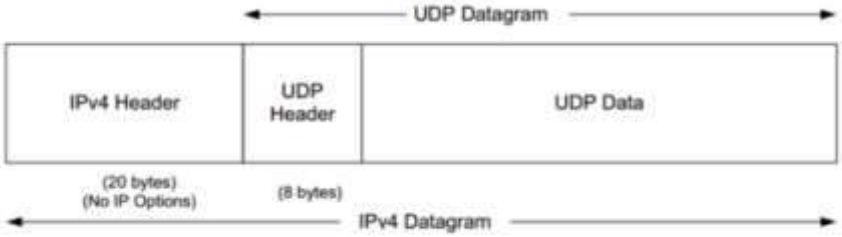


Fig 4.2: IPV4

UDP application

Here are few applications where UDP is used to transmit data:

- Domain Name Services
- Simple Network Management Protocol
- Trivial File Transfer Protocol
- Routing Information Protocol
- Kerberos

Basis	Transmission control protocol (TCP)	User datagram protocol (UDP)
Type of Service	TCP is a connection-oriented protocol. Connection-orientation means that the communicating devices should establish a connection before transmitting data and should close the connection after transmitting the data.	UDP is the Datagram-oriented protocol. This is because there is no overhead for opening a connection, maintaining a connection, and terminating a connection. UDP is efficient for broadcast and multicast types of network transmission.
Reliability	TCP is reliable as it guarantees the delivery of data to the destination router.	The delivery of data to the destination cannot be guaranteed in UDP.
Error checking mechanism	TCP provides extensive error-checking mechanisms. It is because it provides flow control and acknowledgment of data.	UDP has only the basic error checking mechanism using checksums.
Acknowledgment	An acknowledgment segment is present.	No acknowledgment segment.
Sequence	Sequencing of data is a feature of Transmission Control Protocol (TCP). this means that packets arrive in order at the receiver.	There is no sequencing of data in UDP. If the order is required, it has to be managed by the application layer.
Speed	TCP is comparatively slower than UDP.	UDP is faster, simpler, and more efficient than TCP.
Retransmission	Retransmission of lost packets is possible in TCP, but not in UDP.	There is no retransmission of lost packets in the User Datagram Protocol (UDP).
Header Length	TCP has a (20-60) bytes variable length header.	UDP has an 8 bytes fixed-length header.
Weight	TCP is heavy-weight.	UDP is lightweight.
Handshaking Techniques	Uses handshakes such as SYN, ACK, SYN-ACK	It's a connectionless protocol i.e. No handshake
Broadcasting	TCP doesn't support Broadcasting.	UDP supports Broadcasting.
Protocols	TCP is used by HTTP, HTTPs, FTP, SMTP and Telnet.	UDP is used by DNS, DHCP, TFTP, SNMP, RIP, and VoIP.
Stream Type	The TCP connection is a byte stream.	UDP connection is message stream.
Overhead	Low but higher than UDP.	Very low.

MPTCP

Goals for MPTCP are: • It should be capable of using multiple network paths for a single connection. • It must be able to use the available network paths at least as well as regular TCP, but without starving TCP. • It must be as usable as regular TCP for existing applications. • Enabling MPTCP must not prevent connectivity on a path where regular TCP works.

MPTCP manages the underlying TCP connections (called subflows6) that are used to carry the actual data. From an architectural viewpoint, MPTCP acts as a shim layer between the socket interface and one or more TCP subflows, as shown in figure 1. MPTCP requires additional signaling between the end hosts. It achieves this by using TCP options to achieve the following goals: • Establish a new MPTCP connection. • Add subflows to an MPTCP connection. • Transmit data on the MPTCP connection. An MPTCP connection is established by using the three-way handshake with TCP options to negotiate its usage. The MP_CAPABLE option in the SYN segment indicates that the client supports MPTCP. This option also contains a random key used for security purposes. If the server supports MPTCP, then it replies with a SYN+ACK segment that also contains the MP_CAPABLE option. This option contains a random key chosen by the server. The third ACK of the three-way handshake also includes the MP_CAPABLE option to confirm the utilization of MPTCP and the keys to enable stateless servers. The three-way handshake shown in figure 2 creates the first TCP subflow over one interface. To use another interface, MPTCP uses a three-way handshake to establish one subflow over this interface. Adding a subflow to an existing MPTCP connection requires the corresponding MPTCP connection to be uniquely identified on each end host. With regular TCP, a TCP connection is always identified by using the tuple . Unfortunately, because NAT is present, the addresses and port numbers that are used on the client may not be the same as those exposed to the server. Although on each host the 4-tuple is a unique local

identification of each TCP connection, this identification is not globally unique. MPTCP needs to be able to link each subflow to an existing MPTCP connection. For this, MPTCP assigns a locally unique token to each connection. When a new subflow is added to an existing MPTCP connection, the MP_JOIN option of the SYN segment contains the token of the associated MPTCP connection. This is illustrated in figure 3. The astute reader may have noticed that the MP_CAPABLE option does not contain a token. Still, the token is required to enable the establishment of subflows. To reduce the length of the MP_CAPABLE option and avoid using all the limited TCP options space (40 bytes) in the SYN segment, MPTCP derives the token as the result of a truncated hash of the key. The second function of the MP_JOIN option is to authenticate the addition of the subflow. For this, the client and the server exchange random nonces, and each host computes an HMAC (hash-based message authentication code) over the random nonce chosen by the other host and the keys exchanged during the initial handshake. Now that the subflows have been established, MPTCP can use them to exchange data. Each host can send data over any of the established subflows. Furthermore, data transmitted over one subflow can be retransmitted on another to recover from losses. This is achieved by using two levels of sequence numbers.6 The regular TCP sequence number ensures that data is received in order over each subflow and allows losses to be detected. MPTCP uses the data sequence number to reorder the data received over different subflows before passing it to the application.

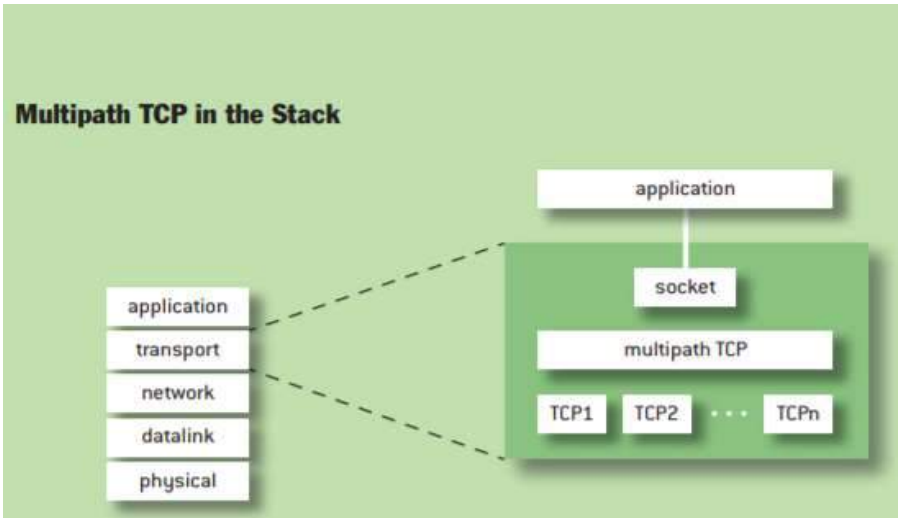


Fig 4.3: Multipath TCP in stack

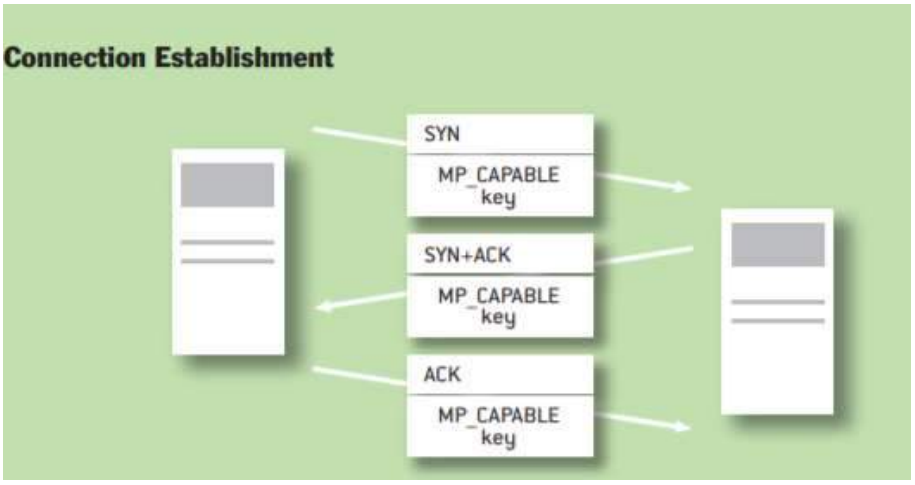


Fig 4.4: Connection Establishment

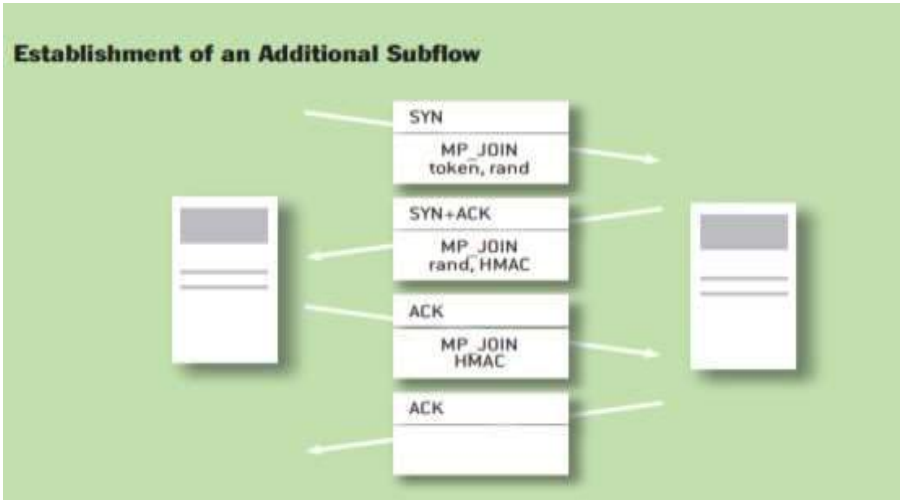


Fig 4.5: Establishment of an additional subflow

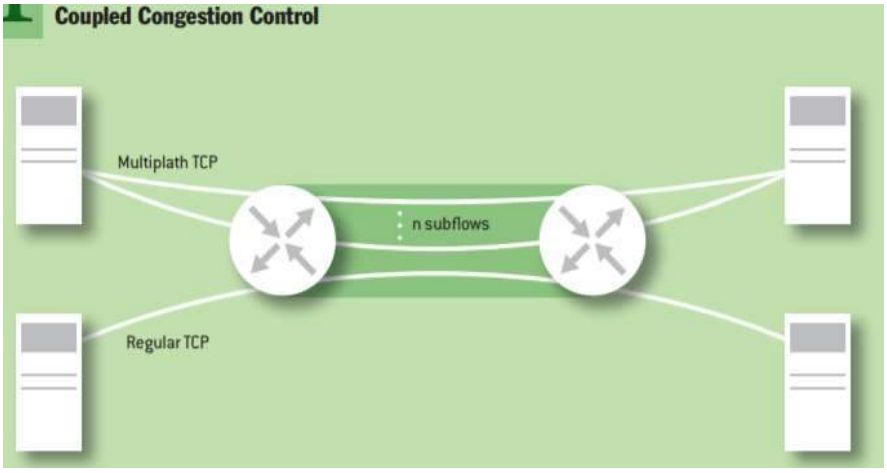


Fig 4.6: Coupled congestion control

From a congestion-control viewpoint, using several subflows for one connection leads to an interesting problem. With regular TCP, congestion occurs on one path between the sender and the receiver. MPTCP uses several paths, and two paths will typically experience different levels of congestion. A naive solution to the congestion problem in MPTCP would be to use the standard TCP congestion-control scheme on each subflow. This can be easily implemented but leads to unfairness with regular TCP. In the network depicted in figure 4, two clients share the same bottleneck link. If the MPTCP-enabled client uses two subflows, then it will obtain two-thirds of the shared bottleneck. SYN MP__JOIN token, rand SYN+ACK MP__JOIN rand, HMAC ACK ACK MP__JOIN HMAC Establishment of an Additional Subflow n subows Multiplath TCP Regular TCP Coupled Congestion Control NETWORKS 5 This is unfair because if this client used regular TCP, it would obtain only half of the shared bottleneck.

DCCP

DCCP provides an unreliable transport service, similar to UDP, but with adaptive congestion control, similar to TCP and Stream Control Transmission Protocol (SCTP). DCCP can be viewed equally well as either UDP-plus-congestion-control or TCP-minus-reliability (although, unlike TCP, DCCP offers multiple congestion control algorithms)

DTLS

Datagram Transport Layer Security (DTLS) is **a protocol used to secure datagram-based communications**. It's based on the stream-focused Transport Layer Security (TLS), providing a similar level of security. ... However, DTLS gains the benefits of datagram protocols, too; in particular, the lower overhead and reduced latency

DTLS over DCCP

The approach here is very straightforward -- DTLS records are transmitted in the Application Data fields of DCCP-Data and DCCP-DataAck packets (in the rest of the document assume that "DCCP-Data packet" means "DCCP-Data or DCCP-DataAck packet"). Multiple DTLS records MAY be sent in one DCCP-Data packet, as long as the resulting packet is within the Path Maximum Transfer Unit (PMTU) currently in force for normal data packets, if fragmentation is not allowed (the Don't Fragment (DF) bit is set for IPv4 or no fragmentation extension headers are being used for IPv6), or within the current DCCP maximum packet size if fragmentation is allowed. A single DTLS record MUST be fully contained in a single DCCP-Data packet; it MUST NOT be split over multiple packets.

DCCP and DTLS Sequence Numbers

Both DCCP and DTLS use sequence numbers in their packets/records. These sequence numbers serve somewhat, but not completely, overlapping functions. Consequently, there is no connection between the sequence number of a DCCP packet and the sequence number in a DTLS record contained in that packet, and there is no connection between sequence number-related features such as DCCP synchronization.

DCCP and DTLS Connection Handshakes

Unlike UDP, DCCP is connection-oriented, and has a connection handshake procedure that precedes the transmission of DCCP-Data and DCCP-DataAck packets. DTLS is also connection-oriented, and has a handshake procedure of its own that must precede the transmission of actual application information. Using the rule of mapping DTLS records to DCCP-Data and DCCP-DataAck packets, the two handshakes are forced to happen in series, with the DCCP handshake first, followed by the DTLS handshake. This is how TLS over TCP works. However, the DCCP handshake packets DCCP-Request and DCCP-Response have Application Data fields and can carry user data during the DCCP handshake, and this creates the opportunity to perform the handshakes partially in parallel. DTLS client implementations MAY choose to transmit one or more DTLS records (typically containing DTLS handshake messages or parts of them) in the DCCP-Request packet. A DTLS server implementation MAY choose to process these records as usual, and if it has one or more DTLS records to send as a response (typically containing DTLS handshake messages or parts of them), it MAY include those records in the DCCP-Response packet. DTLS servers MAY also choose to delay the response until the DCCP handshake completes and then send the DTLS response in a DCCP-Data packet. Note that even though the DCCP handshake is a reliable process (DCCP handshake messages are retransmitted as required if messages are lost), the transfer of Application Data in DCCP-Request and DCCP-Response packets is not necessarily reliable. For example, DCCP server implementations are free to discard Application Data received in DCCP-Request packets. And if DCCP-Request or DCCP-Response packets need to be retransmitted, the DCCP implementation may choose to not include the Application Data present in the initial message.

SCTP Requirements

Number of Inbound and Outbound Streams: An association between the endpoints A and Z provides n streams

from A to Z and m streams from Z to A. A pair consisting of two streams with the same stream identifier is considered and used as one bi-directional stream. Thus an SCTP association can be considered as a set of min(n,m) bi- directional streams and (max(n,m) - min(n,m)) uni-directional streams.

Fragmentation of User Messages

To avoid the knowledge and handling of the MTU inside TLS, SCTP MUST provide fragmentation of user messages, which is an optional feature of [RFC2960]. Since SCTP is a message oriented protocol, it must be able to transmit all TLS records as SCTP user messages. Thus the supported maximum length of SCTP user messages MUST be at least 2^14+ 2048 + 5 = 18437 bytes, which is the maximum length of a TLS Ciphertext, as defined in [RFC2246]. Note that an SCTP implementation might need to support the partial delivery API to be able to support the transport of user messages of this size. Therefore, SCTP takes care of fragmenting and reassembling the TLS records in order to avoid IP-fragmentation.

TLS Requirements

Supported Ciphersuites

A TLS implementation for TLS over SCTP MUST support at least the ciphersuite TLS_RSA_WITH_AES_128_CBC_SHA as defined in [RFC3268].

Connections and Bi-directional Streams: TLS makes use of a bi-directional stream by establishing a connection over it. This means that the number of connections for an association is limited by the number of bi-directional streams. The TLS handshake protocol is used on each bi-directional stream separately. Each handshake can be: - a full handshake or - an abbreviated handshake that resumes a TLS session with a session id from another connection (on the same or another association). After completing the handshake for a connection, the bi-directional stream can be used for TLS-based user data transmission. It should also be noted that the handshakes for the different connections are independent and can be delayed until the bi-directional stream is used for user data transmission.

Usage of bi-directional streams: It is not required that all bi-directional streams are used for TLS- based user data transmission. If TLS is not used, it is called SCTP- based user data transmission.

SCTP-based user data transmission: If a bi-directional stream is not used for TLS-based communication there are no restrictions on the features provided by SCTP for SCTP- based user data transmission.

TLS-based user data transmission: In general, the bi-directional stream will be used for TLS-based user data transmission and it SHOULD NOT be used for SCTP-based user data transmission. The exception to this rule is for protocols which contain upgrade-to-TLS mechanisms, such as those of HTTP upgrade [RFC2817] and SMTP over TLS [RFC3207].

Features and Services	DCCP	SCTP	UDP
Connection Oriented	Yes	Yes	No
Un-ordered Data Delivery	Yes	Yes	Yes
Ordered Data Delivery	No	Yes	No
Reliable	No	Yes	No
Congestion Control	Yes	Yes	No
Flow Control	Optional	Yes	No
Multi-streaming	No	Yes	No
Multi-homing	No	Yes	No

comparison of DCTP,SCTP,UDP

Session Layer Protocols

Most of the IP applications, including IoT applications use TCP or UDP for transport. However, there are several message distribution functions that are common among many IoT applications; it is desirable that these functions be implemented in an interoperable standard ways by different applications. These are the so called "Session Layer" protocols.

MQTT

Message Queue Telemetry Transport (MQTT) was introduced by IBM in 1999 and standardized by OASIS in 2013. It is designed to provide embedded connectivity between applications and middleware's on one side and networks and communications on the other side. It follows a publish/subscribe architecture, as shown in Figure 5 , where the system consists of three main components: publishers, subscribers, and a broker. From IoT point of view, publishers are basically the lightweight sensors that connect to the broker to send their data and go back to sleep whenever possible. Subscribers are applications that are interested in a certain topic, or sensory data, so they connect to brokers to be informed whenever new data are received. The brokers classify sensory data in topics and send them to subscribers interested in the topics.

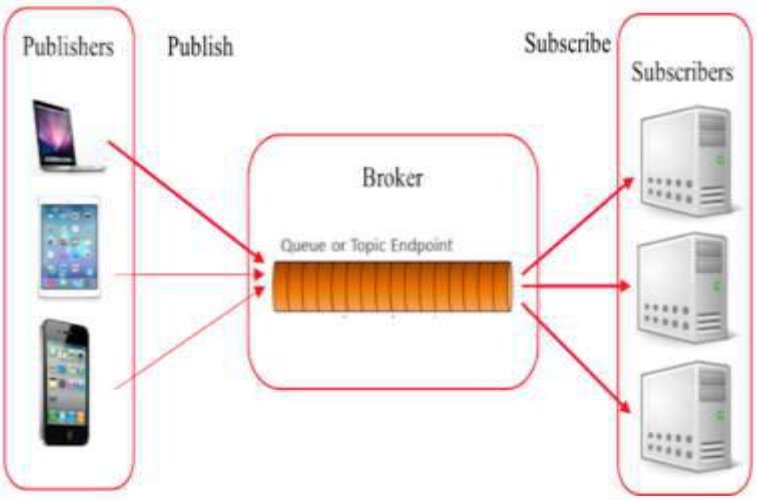


Fig 4.7: MQTT

SMQTT

An extension of MQTT is Secure MQTT (SMQTT) which uses encryption based on lightweight attribute based encryption. The main advantage of using such encryption is the broadcast encryption feature, in which one message is encrypted and delivered to multiple other nodes, which is quite common in IoT applications. In general, the algorithm consists of four main stages: setup, encryption, publish and decryption. In the setup phase, the subscribers and publishers register themselves to the broker and get a master secret key according to their developer's choice of key generation algorithm. Then, when the data is published, it is encrypted, published by the broker which sends it to the subscribers and finally decrypted at the subscribers which have the same master secret key. The key generation and encryption algorithms are not standardized. SMQTT is proposed only to enhance MQTT security feature

AMQP

The Advanced Message Queuing Protocol (AMQP) is another session layer protocol that was designed for financial industry. It runs over TCP and provides a publish/ subscribe architecture which is similar to that of MQTT. The difference is that the broker is divided into two main components: exchange and queues, as shown in Figure 6. The exchange is responsible for receiving publisher messages and distributing them to queues based on pre-defined roles and conditions. Queues basically represent the topics and subscribed by subscribers which will get the sensory data whenever they are available in the queue

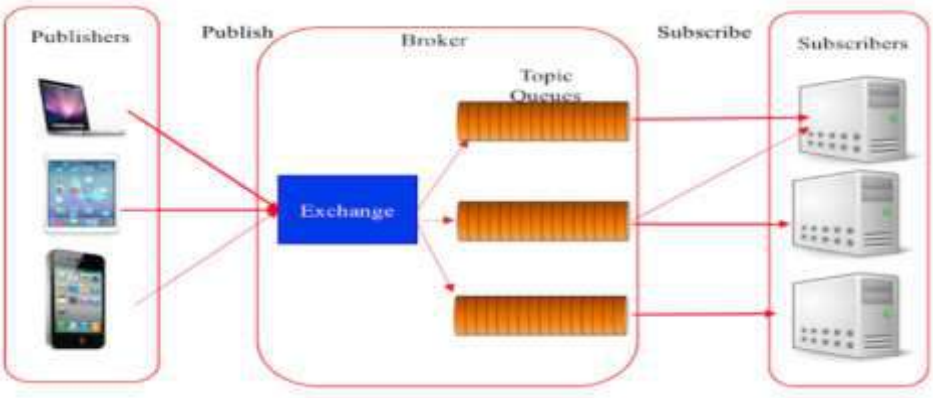


Fig 4.8: AMQP

CoAP

The Constrained Application Protocol (CoAP) is another session layer protocol designed by IETF Constrained RESTful Environment (Core) working group to provide lightweight RESTful (HTTP) interface. Representational State Transfer (REST) is the standard interface between HTTP client and servers. However, for lightweight applications such as IoT, REST could result in significant overhead and power consumption. CoAP is designed to enable low-power sensors to use RESTful services while meeting their power constrains. It is built over UDP, instead of TCP commonly used in HTTP and has a light mechanism to provide reliability. CoAP architecture is divided into two main sublayers: messaging and request/response. The messaging sublayer is responsible for reliability and duplication of messages while the request/response sublayer is responsible for communication. As shown in Figure 7 , CoAP has four messaging modes: confirmable, non-confirmable, piggyback and separate. Confirmable and non-confirmable modes represent the reliable and unreliable transmissions, respectively while the other modes are used for request/response. Piggyback is used for client/server direct communication where the server sends its response directly after receiving the message, i.e., within the acknowledgment message. On the other hand, the separate mode is used when the server response comes in a message separate from the acknowledgment, and may take some time to be sent by the server. As in HTTP, CoAP utilizes GET, PUT, PUSH, DELETE messages requests to retrieve, create, update, and delete, respectively

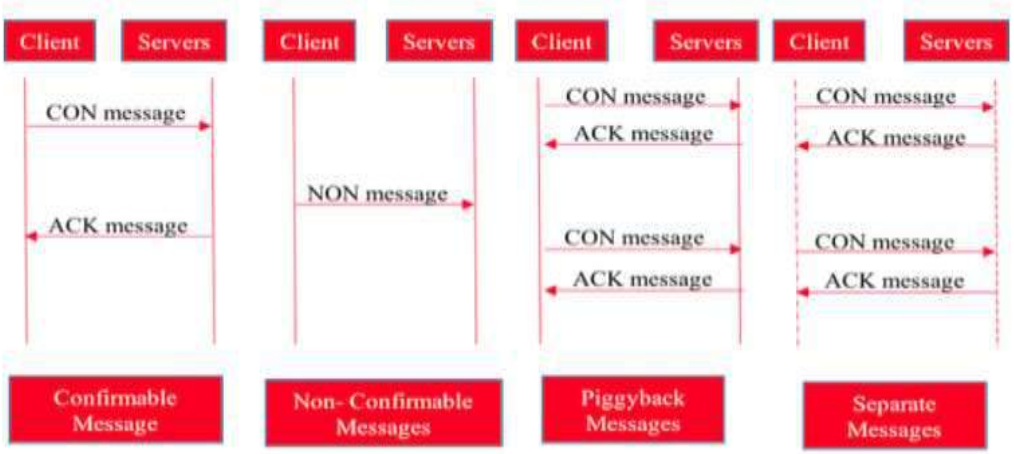


Figure4.9 COAP messages

XMPP

Extensible Messaging and Presence Protocol (XMPP) is a messaging protocol that was designed originally for chatting and message exchange applications. It was standardized by IETF more than a decade ago. Hence, it is well known and has proven to be highly efficient over the internet. Recently, it has been reused for IoT applications as well as a protocol for SDN. This reusing of the same standard is due to its use of XML which makes it easily extensible. XMPP supports both publish/ subscribe and request/ response architecture and it is up to the application developer to choose which architecture to use. It is designed for near real-time applications and, thus, efficiently supports low-latency small messages. It does not provide any quality of service guarantees and, hence, is not practical for M2M communications. Moreover, XML messages create additional overhead due to lots of headers and tag formats which increase the power consumption that is critical for IoT application. Hence, XMPP is rarely used in IoT but has gained some interest for enhancing its architecture in order to support IoT applications

DDS

Data Distribution Service (DDS) is another publish/subscribe protocol that is designed by the Object Management Group (OMG) for M2M communications [DDS]. The basic benefit of this protocol is the excellent quality of service levels and reliability guarantees as it relies on a broker-less architecture, which suits IoT and M2M communication. It offers 23 quality-of-service levels which allow it to offer a variety of quality criteria including: security, urgency, priority, durability, reliability, etc. It defines two sublayers: data-centric publish- subscribe and data-local reconstruction sublayers. The first takes the responsibility of message delivery to the subscribers while the second is optional and allows a simple integration of DDS in the application layer. Publisher layer is responsible for sensory data distribution. Data writer interacts with the publishers to agree about the data and changes to be sent to the subscribers. Subscribers are the receivers of sensory data to be delivered to the IoT application. Data readers basically read the published data and deliver it to the subscribers and the topics are basically the data that are being published. In others words, data writers and data reader take the responsibilities of the broker in the broker-based architectures.

Protocols	UDP/TCP	Architecture	Security and QoS	Header Size (bytes)	Max Length(bytes)
MQTT	TCP	Pub/Sub	Both	2	5
AMQP	TCP	Pub/Sub	Both	8	-
CoAP	UDP	Req/Res	Both	4	20 (typical)
XMPP	TCP	Both	Security	-	-
DDS	TCP/UDP	Pub/Sub	QoS	-	-

TEXT / REFERENCE BOOKS

1 Jan Holler, VlasiosTsiatsis, Catherine Mulligan, Stefan Avesand, StamatisKarnouskos, David Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence Press, 2014.

2 Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.

3 Hassan, Qusay F., ed. Internet of Things A to Z: technologies and applications. John Wiley & Sons, 2018.

4 Holler, Jan, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand, and David Boyle. Internet of Things. Academic Press, 2014.

5 Hersent, Olivier, David Boswarthick, and Omar Elloumi. The internet of things: Key applications and protocols. John Wiley & Sons, 2011.

6 Bernd Scholz- -3-642-19156-5 e-ISBN Architecting the Internet of Things, ISBN 978-3-642-19156-5, 978-3-642-19157-2, Springer

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – I – IoT Architecture and its Protocols– SCSA1408

UNIT 5 SERVICE LAYER PROTOCOLS & SECURITY

Service Layer one M2M, ETSI M2M, OMA, BBF Security in IoT Protocols MAC 802.15.4 , 6LoWPAN, RPL, Application Layer

Service Layer one M2M

Functional Building Blocks This Common M2M Service Layer should be agnostic to underlying network technology (yet leveraging the unique features of these underlying networks), and it will use network layer services (such as security (encryption and authentication), QoS, policy, provisioning, etc.) through an adaptation layer/APIs. These Common M2M Service Layer functions include: • Provisioning: On board new devices by creating new subscription in the common M2M service layer and network layer; activate/deactivate/suspend/resume network and service subscriptions. • Device Management: Manage all aspects of the devices including configuration, firmware upgrades, application lifecycle management, device lock and wipe.

Security: Generate relevant key material for secure communications; Authenticate devices before they can register and modify resources. Prevent unauthorized entities from sending IP packets to the devices. Provide a secure connection to the device

- **Application and Device Registrations**: Application and devices will be able to register with the service layer entity for various services. Registration will involve authentication or verification of credentials and creation or allocation of resources within the server and the database. A profile with the capabilities of the device and the type of services allowed for the applications are created.
- **Resource Management:** Applications and devices will be able to create, update, and delete resource objects containing various attributes in the service layer. Entities will be able to discover resources.
- **Content push/pull Services:** Provide API for applications to perform unicast and multicast data push to specified devices within the specified time window. Push may be result of a notification that is triggered as a result of modification of a resource. Provide API for applications to pull data from one or more devices within the specified time window or specified periodicity or other policies that have been established.
- **Store and Forward Messaging:** Applications may request messages to be sent to one or more devices that may not be registered with the network at that time. In this case the communications management entity shall store and aggregate the messages and forward them to the devices at a later time when the devices wake up.
- **Protocol Translation:** Translate protocols between application and device as needed. For example, applications may use HTTP while devices may use Constrained Application Protocol (CoAP) or Zigbee® Smart Energy 2.0 protocol.
- **Subscribe/Notification:** Application and devices should be able to subscribe to receive notifications upon certain events or when certain resources are updated. Events may be specified as rules on certain resource data.
- **Policy Framework**: Framework to establish and incorporate in the session orchestration, data aggregation and storage, the network provider and application provider policies. Examples include incorporating a location tag or time stamp on all data, policy restricting sessions only to certain hours of the day.
- **Location and Geo Fencing:** Provide device and network based location and location related services such as creating a geo-fence or identifying a group of devices within a region or adding a location tag to the device data.

- **Groups Management:** Framework for creation of groups by specifying the members of the group through one of the identifiers of the device, adding additional members or removing members; setting group attributes.
- **Device Triggering:** Provide the capability to trigger the device to register with the network and an application through a secondary means such as an SMS. Provide information about the status of the device in the network.
- **Access Control:** Control the access to the data collected from the devices based on access restrictions specified by applications in terms which users or devices can access what resources.
- **Data Processing and Storage:** Provide temporary and permanent storage for data collected from devices. Process queries on data collected. Provide threshold and expression rules setting and execution on the various data collected from the devices. Notifications could be triggered based on the outcome of the rules testing.
- **Consumption Statistics/records:** Process queries regarding the usage of network resources by a device or a group of devices for billing reconciliation.
- **API Management:** Manage API usage, such as authentication and authorization of calls to APIs provided.

The functions above have been identified as essential by the following verticals: Connected Vehicle; Smart Grid; eHealth; and Connected Home. The goal of this diligence was to determine which Common M2M service layer functions were necessary for their respective vertical segment and to then converge on those common functions as the components for a "Common M2M Service Layer" capability

The functions proposed for a Common M2M Service Layer are:

Device Management

• Provision/Activate (Individual and bulk) and Bootstrap

• Suspend/Resume

• Configuration Management

• Firmware/Software Management

• Inventory Management

• Diagnostics (resource information, status)

Policy & Resource Management

• Authentication and Registration (Identity Management)

• Establish communications session (Add/Delete/Modify)

• QoS/SLA for communication session

• Billing, Charging, and Rating rules

• Group Management

• Security Management (Data confidentiality, integrity, abuse prevention, privacy)

API Services

• Definition, Authentication/Authorization and Security

• Service to Device (Management, Establish/Teardown Communication Flows)

• Service to Policy/Resource Management (Rx Extensions for Group Management)

• Service to Data/Metadata Management (Storage/Retrieval)

• Service to Applications (Management, Communications Flows)

Data/Metadata Management

• Data processing and append (location, timestamp)

• Data storage/retrieval

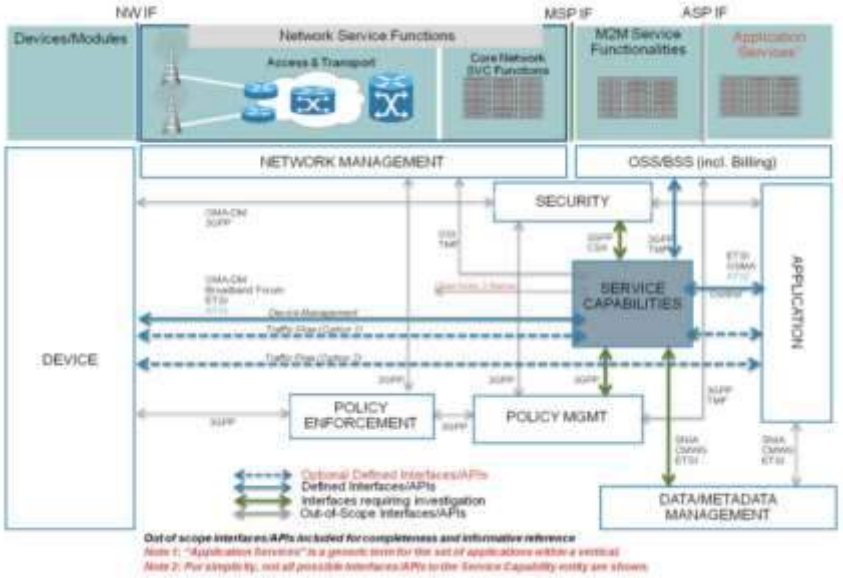


Fig: 5.1 Common M2M Service Layer

ETSI M2M architecture

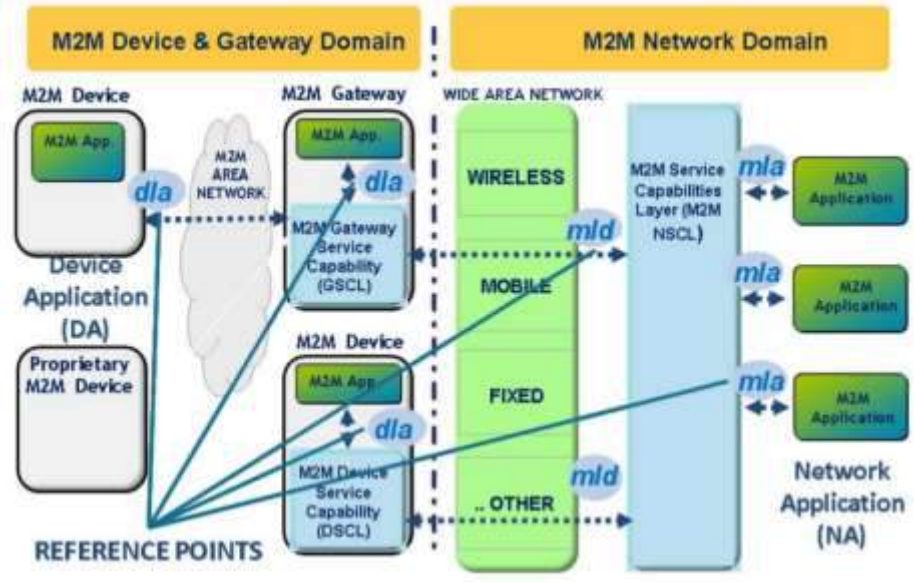


Fig: 5.2 M2M Functional Architecture Overview

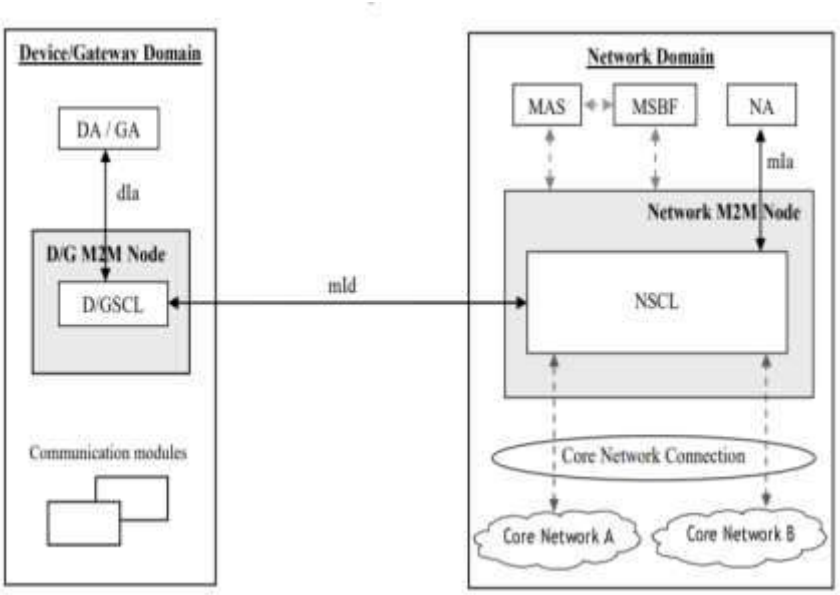


Fig: 5.3 M2M Architecural framework

M2M Service Capabilities Layer provides functions that are to be exposed on the reference points. M2M SCs can use

Core Network functionalities through a set of exposed interfaces (e.g. existing interfaces specified by 3GPP, 3GPP2,

ETSI TISPAN, etc.). Additionally, M2M SCs can interface to one or several Core Networks.

the following terms will be used are :

• NSCL: Network Service Capabilities Layer refers to M2M Service Capabilities in the Network Domain.
• GSCL: Gateway Service Capabilities Layer refers to M2M Service Capabilities in the M2M Gateway.
• DSCL: Device Service Capabilities Layer refers to M2M Service Capabilities in the M2M Device.
• SCL: Service Capabilities Layer refers to any of the following: NSCL, GSCL, or DSCL.
• D/G SCL: refers to any of the following: DSCL, GSCL.

The external reference points (mIa, mId, dIa) are mandated and are required for ETSI M2M compliance.

M2M Node: is a logical representation of the M2M components in the M2M Device, M2M Gateway, or the M2M Core. An M2M Node shall include one SCL, and optionally an M2M Service Bootstrap function and an M2M Service Connection function. An M2M Node relies on a Secured Environment Domain, controlled by the M2M Service Provider associated with the SCL, to protect Sensitive Functions and Sensitive Data. A Device/Gateway M2M Node shall be instantiated upon pre-provisioning or executing an M2M Service Bootstrap procedure on the M2M Device/Gateway with an M2M Service Provider. Each Device/Gateway M2M Node may be instantiated with only one M2M Service Provider.

M2M Applications: are respectively Device Application (DA), Gateway Application (GA) and Network Application (NA). DA could reside in an M2M Device which implements M2M Service Capabilities or alternatively reside in an M2M Device which does not implement M2M Service.

OMA

OMA Lightweight M2M (LwM2M) is a protocol from the Open Mobile Alliance for machine to machine (M2M) or Internet of things (IoT) device management and service enablement. It offers an approach for managing IoT devices and allows devices and systems from different vendors to co-exist in an IoT- ecosystem.

Management: Remote Entity Management (REM) service capability shall be supported by all M2M SCLs. Both OMA-DM and BBF TR-069 are considered as the supporting enablers to be reused in ETSI M2M

functional architecture to implement the REM service capability. An M2M system may choose to implement either or both of them. Other device management enablers may also be reused in a similar manner, but the details are out of scope. The mIa reference point shall support the RESTful interface procedures to allow the NREM to handle the request from M2M Network Applications for the purpose of the remote entity management

The mId reference point shall support the RESTful interface procedures to allow the D/GREM to create and/or resources in the NREM for the purpose of the remote entity management thereafter. It shall also support corresponding OMA-DM [i.2]/TR-069 [i.3] protocol interfaces and procedures for managing M2M Devices/Gateways enabled by OMA-DM [i.2]/TR-069 [i.3].

A <mgmtObj> or <mgmtCmd> resource in the NSCL represents either:

• high-level management functionalities (e.g. ETSI M2M specific data model) which shall be supported by the underlying Management Object(s) on the remote entity; or

• low-level functionalities on the remote entity mapped from the data model as specified by existing device management technologies (e.g. OMA-DM, BBF TR-069 [i.3]).

Through the manipulation of <mgmtObj> or <mgmtCmd> resources, ETSI M2M REM supports the following management functions at different layers of remote entities:

• M2M application lifecycle management: installing, removing and upgrading applications in an M2M Device/M2M Gateway.

• M2M service management: configuration management for the M2M Service Capabilities in the M2M Device/M2M Gateway.

• M2M Area Network management: configuration management for the M2M Area Networks (namely Capillary Network).

• M2M device management: configuration management of the M2M Device/M2M Gateway.

Management Protocol OMA DM: The OMA DM is a management protocol of mobile devices which makes possible to remotely reach and control devices resources. This protocol is conceived for small devices such as Mobile phones, Smartphones, Tablets, robust laptops, mobile printers, PDAs (Personal DIGITAL Assistant), etc. OMA DM is for sensitive devices, in other words, devices which have a limited storage memory or limited communication bandwidth, like a wireless connectivity 4. It is difficult for the server to manage mobiles because of their specific configuration and their particular characteristicswhich depend on each manufacturer. The OMA DM protocol proposed a standard framework making it possible forthe suppliers to define the description of the devices and to communicate it to the server. The server is based on this description to send operations. This description is called the management tree of the device. This tree organizes allthe managed objects of the device in which each node is addressed (in a single way) by an URI (Universal ResourceIdentifier). The server is so able, if he has the access right, to modify the management tree by sending operations Get,Add, Delete and Update. The OMA DM protocol consists in exchanging a sequence of XML messages between theserver and the client, more particularly the subset defined by SyncML (Synchronization Markup Language). SyncML contains a set of well-defined messages which are transmitted between a server and a client taking part in an operation of synchronization. The communication is carried out in two phases; the phase of initiation then the phase of management of the device.

Management protocol BBF TR-069BBF: TR-069 is a protocol for remote management of end-user devices. He offers to the user a set of administration services, control and diagnosis while avoiding the problems involved in material and software diversities. The standard TR-069 was developed for the automatic configuration of the devices with Internet access with the automatic configuration servers ACS (modem, router, Gateway, Set-Top Boxings, VoIP-telephone, etc.). Since management is done by recovery of the values of devices parameters, the latter are organized according to a hierarchical well-defined structure which is more or less common to all models of devices and manufacturers. This model is published into two formats:• XML

format where each model contains a detailed description of devices as well as the modifications carried out.• Pdf version container of information readable by human. This last format is seldom used, since it requires a human intervention which is not always practical.

	OMA DM	BBF TR-069
Data model	Tree	XML file
Device type	Mobile Devices	Network Devices
Trigger	The server informs Device by SMS or WAP Push	The manager connects to the device embeded http server
Architecture Style	Uses the style of architecture REST	Non RESTful management command, RPC

Fig : 5.4 Comparison table of OMA and BBF

The manipulation of files is done with a set of operations; “GetParameterNames”, “SetParametersValues”, “GetPa-rametersAttributes”, “AddObject”, “DeleteObject”, etc. Even if the list of the parameters and their attributes are welldefined, most devices do not respect the standards completely. The most current problems are the missing parame-ters. This protocol uses SOAP (Simple Object Access Protocol) to exchange XML messages through RPC (RemoteProcedure Call) which enables two applications to do procedure calls one on the other. The communication is carriedout in two phases; the phase of initiation then the phase of management of the device

Managing M2M Device and M2M Gateway

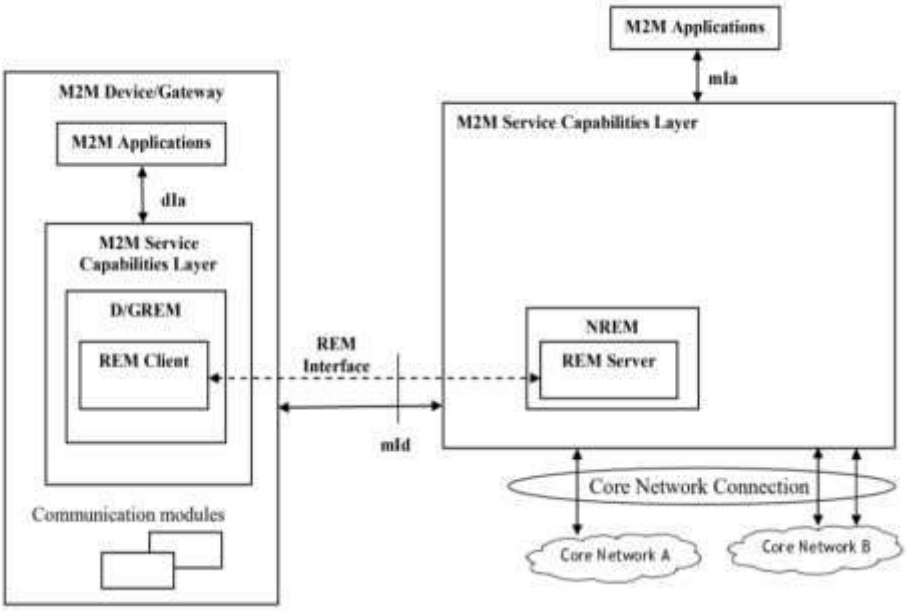


Fig: 5.5 Reference Architecture for managing M2M Device/Gateway

As shown in figure the Device Management Client is integrated as a part of the D/GREM Service Capability, while the Device Management Server is integrated as a part of the NREM Service Capability.

NOTE 1: Alternatively, to REM Server being integrated as a part of the NREM a REM Server may be external to the NREM but interface with the NREM via an implementation-specific interface exposed by the REM Server.

NOTE 2: When ETSI M2M architecture is deployed over 3GPP network, the OMA-DM Server in the 3GPP network layer may be integrated for the purpose of remote management**.**

On the M2M Device and M2M Gateway side: • The D/GREM may collect the Management Object data from the REM Client in the local M2M Device/Gateway and create/update the corresponding and/or resource(s) in the NSCL via the mId reference point. • Any device management activity (e.g. firmware update, fault management) in the Device/Gateway is carried out by the REM Client, communicating with a REM Server in NREM via existing Device Management interfaces.

On the NSCL side: • The NREM triggers OMA-DM or BBF TR-069 Device Management procedures over mId resulting from actions on the or resources by M2M Network Applications via mIa or by M2M Management Functions.

ETSI M2M	OMA DM	BBF TR069
REM Server	DM Server	ACS
REM Client	DM Client	CPE
REM Interface (over mId)	DM-1, DM2	TR069-CWMP

Fig :5.6 Mapping between ETSI M2M and OMA-DM/TR-069 entities

Integration of OMA DM protocol in M2M ETSI platform: OMA DM protocol acts on one of the resources in the resources tree to be able to fulfill its functions. This resource is named <mgmtOb j >. Therefore, it is a question of making a mapping between the two trees; the sub tree of theresource <mgmtOb j >of ETSI M2M and the management tree OMA DM. The strong point of this platform, is that the M2M NA is not supposed to know about the OMA DM protocol mechanism. It just sends a normalized request to receive at the end a normalized answer. Whereas behind this answer, there was a conversion to OMA DM protocol. The request in fact is transformed into OMA DM request and the OMA DManswer is transformed into ETSI format before sending. Fig provides the architecture that is used to build the ETSI M2M service platform. In this figure, we see two service layers. The OMA DM client is integrated in the Device or the Gateway service capabilities layer (SCL) in D/GREM serviceand the OMA DM server is integrated in the Network SCL in the NREM service. The NA sends an ETSI M2M request to manage a device and this by acting on the resource <mgmtOb j >in theresources tree. The request arrives at service NREM and will be mapped in OMA DM format and redirected towards OMA DM server. The OMA DM protocol is a soft protocol, the client is not always listening for requests. So, all requests are stored in the server. Periodically, the client connects to respond to pending requests, this is the initiation phase. Then, the server sends the pending request that will be executed according to the OMA DM protocol in the OMA DM client who resides in the D/GREM. This execution will handle the OMA DM tree of the device to manage it which will generate an answer thereafter. The OMA DM answer will be sent to the OMA DM server to close the management session. Once the spot of management carried out, the NREM will execute this update in the same way on the resource <mgmtOb j >and will end up sending an ETSI answer to the application

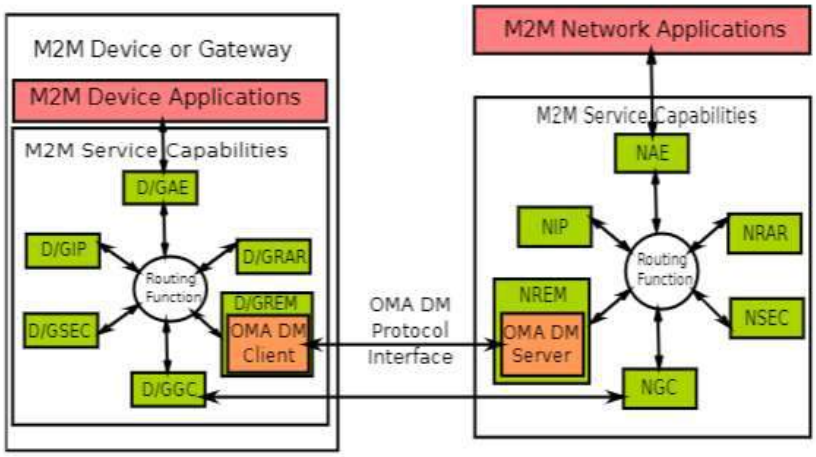


Fig: 5.7 Integration of OM-DM protocol

Security in IoT Protocols

Security is another aspect of IoT applications which is critical and can be found in all almost all layers of the IoT protocols. Threats exist at all layers including datalink, network, session, and application layers.

MAC 802.15.4

MAC 802.15.4 offers different security modes by utilizing the "Security Enabled Bit" in the Frame Control field in the header. Security requirements include confidentiality, authenticapapertion, integrity, access control mechanisms and secured Time-Synchronized Communications.

6LoWPAN

lo6LoWPAN by itself does not offer any mechanisms for security. However, relevant documents include discussion of security threats, requirement and approach to consider in IoT network layer. For example, RFC 4944 discusses the possibility of duplicate EUI-64 interface addresses which are supposed to be unique [RFC4944]. RFC 6282 discusses the security issues that are raised due to the problems introduced in RFC 4944 [RFC6282]. RFC 6568 addresses possible mechanisms to adopt security within constrained wireless sensor devices [RFC6568]. In addition, a few recent drafts in [6Lo] discuss mechanisms to achieve security in 6loWPAN.

RPL

RPL offers different level of security by utilizing a "Security" field after the 4-byte ICMPv6 message header. Information in this field indicates the level of security and the cryptography algorithm used to encrypt the message. RPL offers support for data authenticity, semantic security, protection against replay attacks, confidentiality and key management. Levels of security in RPL include Unsecured, Preinstalled, and Authenticated. RPL attacks include Selective Forwarding, Sinkhole, Sybil, Hello Flooding, Wormhole, Black hole and Denial of Service attacks.

Application Layer

Applications can provide additional level of security using TLS or SSL as a transport layer protocol. In addition, end to end authentication and encryption algorithms can be used to handle different levels of security as required.

IoT Application Layer Protocols

When considering constrained networks and/or a large-scale deployment of constrained nodes, verbose web-

based and data model protocols, may be too heavy for IoT applications. To address this problem, the IoT industry is working on new lightweight protocols that are better suited to large numbers of con- strained nodes and networks. Two of the most popular protocols are CoAP and MQTT. Figure highlights their position in a common IoT protocol stack.

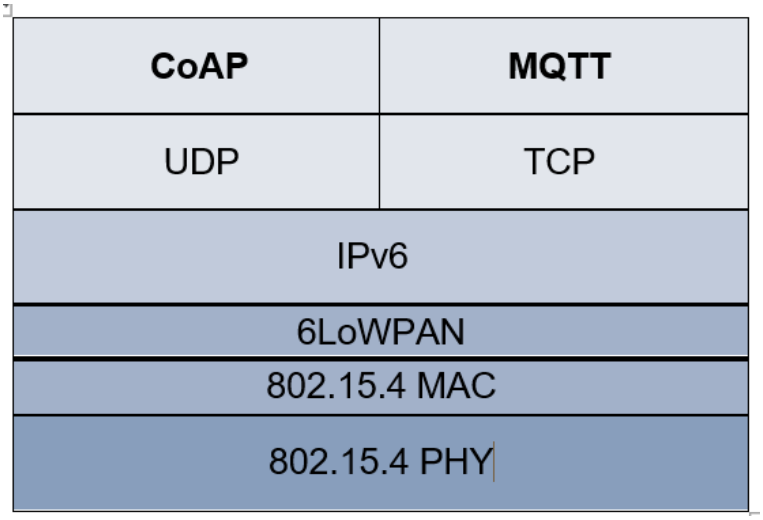


Fig 5.7 Example of a High-Level IoT Protocol Stack for CoAP and MQTT

Constrained Application Protocol (CoAP) resulted from the IETF Constrained RESTful Environments (CoRE) working group's efforts to develop a generic framework for resource-oriented applications targeting constrained nodes and networks. (For more information on the IETF CoRE working group

The CoAP framework defines simple and flexible ways to manipulate sensors and actuators for data or device management. The IETF CoRE working group has published multiple standards-track specifications for CoAP, including the following:

■RFC 6690: Constrained RESTful Environments (CoRE) Link Format
■RFC 7252: The Constrained Application Protocol (CoAP)
■RFC 7641: Observing Resources in the Constrained Application Protocol (CoAP)
■RFC 7959: Block-Wise Transfers in the Constrained Application Protocol (CoAP)
■RFC 8075: Guidelines for Mapping Implementations: HTTP to the Constrained Application Protocol (CoAP)

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). (UDP is discussed earlier in this chapter.) The IETF CoRE working group is studying alternate transport mechanisms, including TCP, secure TLS, and WebSocket. CoAP over Short Message Service (SMS) as defined in Open Mobile Alliance for Lightweight Machine-to-Machine (LWM2M) for IoT device manage- ment is also being considered. (For more information on the Open Mobile Alliance, see http://openmobilealliance.org.)

RFC 7252 provides more details on securing CoAP with DTLS. It specifies how a CoAP endpoint is provisioned with keys and a filtering list. Four security modes are defined: NoSec, PreSharedKey, RawPublicKey, and Certificate. The NoSec and RawPublicKey implementations are mandatory.

From a formatting perspective, a CoAP message is composed of a short fixed-length Header field (4 bytes), a variable-length but mandatory Token field (0–8 bytes), Options fields if necessary, and the Payload field. Figure details the CoAP message format, which delivers low overhead while decreasing parsing complexity.

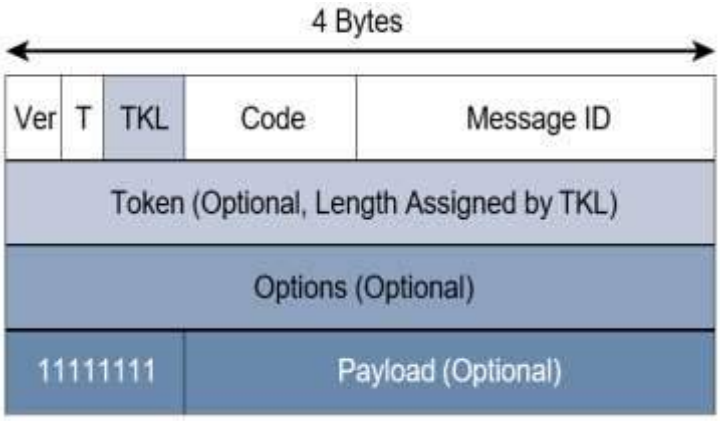


Fig 5.8 CoAP Message Format

In Figure, the CoAP message format is relatively simple and flex- ible. It allows CoAP to deliver low overhead, which is critical for constrained networks, while also being easy to parse and process for constrained devices.

CoAP Message Fields

CoAP Message Field	**Description**
Ver (Version)	Identifies the CoAP version.
T (Type)	Defines one of the following four message types: Confirmable (CON), Non-confirmable (NON), Acknowledgement (ACK), or Reset (RST). CON and ACK are highlighted in more detail in Figure 6-9.
TKL (Token Length)	Specifies the size (0–8 Bytes) of the Token field.
Code	Indicates the request method for a request message and a response code for a response message. For example, in Figure 6-9, GET is the request method, and 2.05 is the response code. For a complete list of values for this field, refer to RFC 7252.
Message ID	Detects message duplication and used to match ACK and RST message types to Con and NON message types.
Token	With a length specified by TKL, correlates requests and responses.
Options	Specifies option number, length, and option value. Capabilities provided by the Options field include specifying the target resource of a request and proxy functions.
Payload	Carries the CoAP application data. This field is optional, but when it is present, a single byte of all 1s (0xFF) precedes the payload. The purpose of this byte is to delineate the end of the Options field and the beginning of Payload.

Message Queuing Telemetry Transport (MQTT)

At the end of the 1990s, engineers from IBM and Arcom (acquired in 2006 by Eurotech) were looking for a reliable, lightweight, and cost-effective protocol to monitor and control a large number of sensors and their data from a central server location, as typi- cally used by the oil and gas industries. Their research resulted in the development and implementation of the Message Queuing Telemetry Transport (MQTT) protocol that is now standardized by the Organization for the Advancement of Structured Information Standards (OASIS). (For more information on OASIS, see www.oasis-open.org.)

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, as shown in Figure

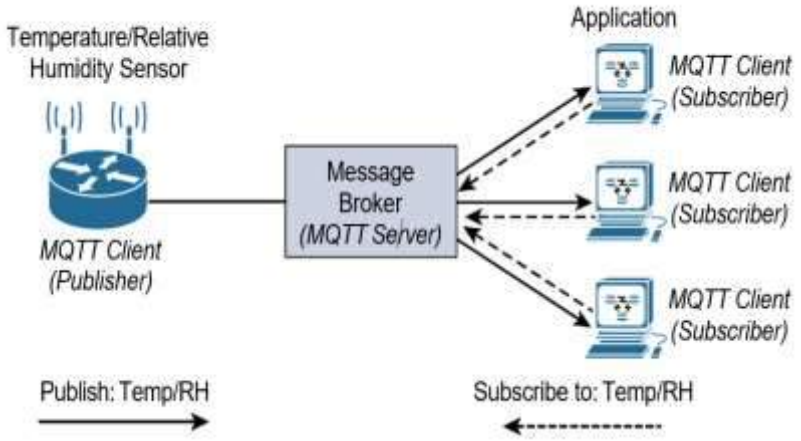


Fig 5.9 MQTT Publish/Subscribe Framework

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 the example illustrated in Figure, the MQTT client on the left side is a temperature (Temp) and relative humidity (RH) sen- sor 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. A great example is the collaboration and social networking application Twitter. With MQTT, clients can subscribe to all data (using a wildcard character) or specific data from the information tree of a publisher. In addition, the presence of a message broker in MQTT decouples the data transmission between clients acting as publishers and sub- scribers. 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. This also means that publishers and subscribers do not have to be online at the same time. MQTT control packets run over a TCP transport using port 1883. TCP ensures an ordered, lossless stream of bytes between the MQTT client and the MQTT server. Optionally, MQTT can be secured using TLS on port 8883, and WebSocket (defined in RFC 6455) can also be used. MQTT is a lightweight protocol because each control packet consists of a 2-byte fixed header with optional variable header fields and optional payload. You should note that a control packet can contain a payload up to 256 MB. Figure provides an overview of the MQTT message format.

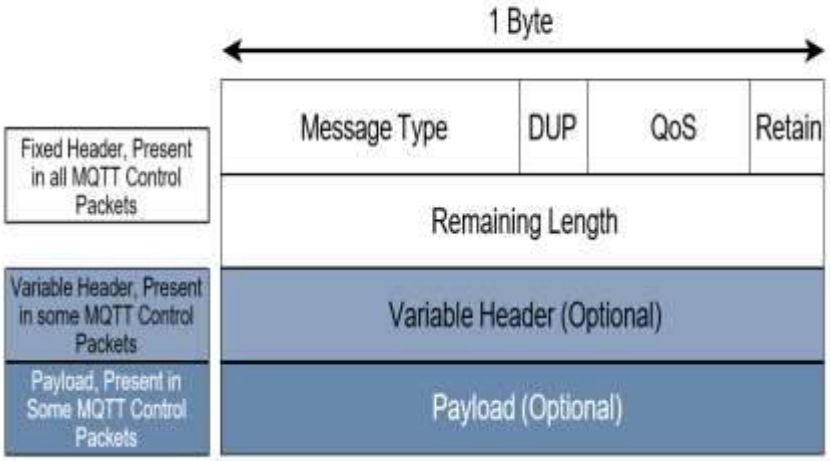


Fig 5.10 MQTT Message Format

Comparison Between CoAP and MQTT

Factor	CoAP	MQTT
Main transport protocol	UDP	TCP
Typical messaging	Request/response	Publish/subscribe
Effectiveness in LLNs	Excellent	Low/fair (Implementations pairing UDP with MQTT are better for LLNs.)
Security	DTLS	SSL/TLS
Communication model	One-to-one	many-to-many
Strengths	Lightweight and fast, with low overhead, and suitable for constrained networks; uses a RESTful model that is easy to code to; easy to parse and process for constrained devices; support for multicasting; asynchronous and synchronous messages	TCP and multiple QoS options provide robust communications; simple management and scalability using a broker architecture
Weakness	Not as reliable as TCP-based MQTT, so the application must ensure reliability.	Higher overhead for constrained devices and networks; TCP connections can drain low-power devices; no multicasting support

Security

The IEEE 802.15.4 specification uses Advanced Encryption Standard (AES) with a 128-bit key length as the base encryption algorithm for securing its data. Established by the US National Institute of Standards and Technology in 2001, AES is a block cipher, which means it operates on fixed-size blocks of data. The use of AES by the US govern- ment and its widespread adoption in the private sector has helped it become one of the most popular algorithms used in symmetric key cryptography. (A *symmetric key* means that the same key is used for both the encryption and decryption of the data.) In addition to encrypting the data, AES in 802.15.4 also validates the data that is sent. This is accomplished by a message integrity code (MIC), which is calculated for the entire frame using the same AES key that is used for encryption.

Enabling these security features for 802.15.4 changes the frame format slightly and consumes some of the payload. Using the Security Enabled field in the Frame Control portion of the 802.15.4 header is the first step to enabling AES encryption. This field is a single bit that is set to 1 for security. Once this bit is set, a field called the Auxiliary Security Header is created after the Source Address field, by stealing some bytes from the Payload field. Figure 4-8 shows the IEEE 802.15.4 frame format at a high level, with the Security Enabled bit set and the Auxiliary Security Header field present.

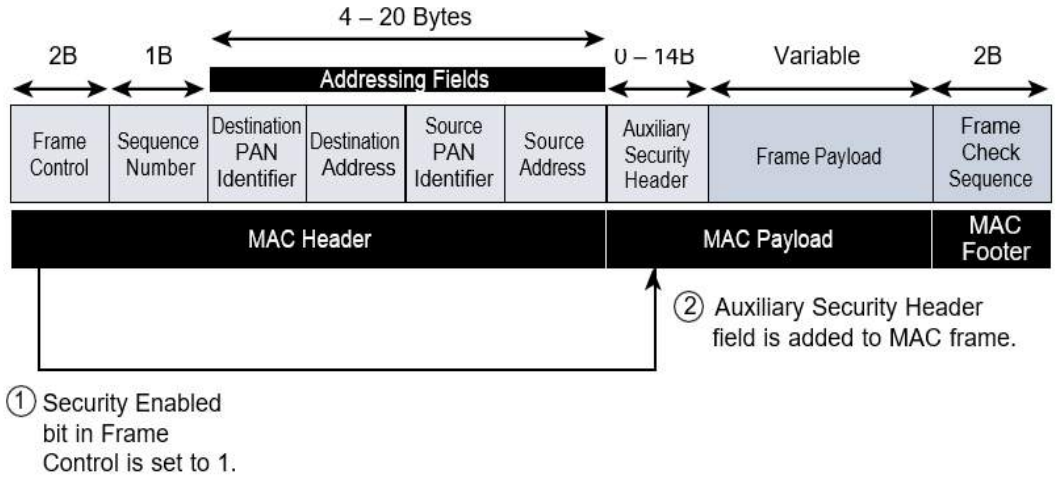


Figure 5.11 :Frame Format with the Auxiliary Security Header Field for 802.15.4-2006 and Later Versions

Security in IEEE 802.15.4 The IEEE 802.15.4-2011 standard provides security services at the MAC layer that, despite being designed to secure communications at the link layer, are valuable in supporting security mechanisms designed at higher layers of the protocol stack illustrated in Fig. This is motivated by the support of efficient symmetric cryptography at the hardware in IEEE 802.15.4 sensing platforms. For example, current sensing platforms employing the cc2420 single-chip RF transceiver from Texas Instruments, as the TelosB mote from Crossbow, support IEEE 802.15.4 security and symmetric cryptography at the hardware using the Advanced Encryption Standard (AES)

Security Modes in The IEEE 802.15.4 Standard

Security mode	Security provided
No Security	Data is not encrypted Data authenticity is not validated
AES-CBC-MAC-32	Data is not encrypted Data authenticity using a 32-bit MIC
AES-CBC-MAC-64	Data is not encrypted Data authenticity using a 64-bit MIC
AES-CBC-MAC-128	Data is not encrypted Data authenticity using a 128-bit MIC
AES-CTR	Data is encrypted Data authenticity is not validated
AES-CCM-32	Data is encrypted Data authenticity using a 32-bit MIC
AES-CCM-64	Data is encrypted Data authenticity using a 64-bit MIC
AES-CCM-128	Data is encrypted Data authenticity using a 128-bit MIC

Security Modes: The IEEE 802.15.4 standard support various security modes at the MAC layer, which are described in Table. The available security modes are distinguished by the security guarantees provided and by the size of the integrity data employed. Fig. illustrates the application of security to an IEEE 802.15.4 link-layer data frame. A protected frame is identified by the Security Enabled Bit field of the Frame Control field being set at the beginning of the header. The Auxiliary Security Header is employed only when security is used, and identifies how security is applied to the frame. In the Auxiliary Security Header, the Security Control field identifies the Security Level mode from the modes identified in Table I, and how the cryptographic key required to process security for the link-layer frame is to be determined by the sender and receiver. The standard employs 128-bit keys that may be known implicitly by the two communication parties, or on the other end determined from information transported in the Key Source and Key Index subfields of the Key Identifier field. The Key Source subfield specifies the group key originator, and the Key Index subfield identifies a key from a specific source. The various security modes require the transportation of security-related information in different configurations, as in Fig. In our following discussion we identify how fundamental security requirements are assured by security at the MAC. Confidentiality: Security as currently defined by IEEE 802.15.4 is optional, given that an application may opt for no security or for security at others layers of the protocol stack. For applications requiring only confidentiality of link layer communications, the transmitted data may be encrypted using AES in the Counter (CTR) mode, using the AES-CTR security mode. As with all the security modes available at the IEEE 802.15.4 MAC layer, 128-bit keys are used to support this requirement. Data Authenticity and Integrity: Applications requiring authenticity and integrity of link-layer communications may use one of the security modes employing AES in the Cypher Block Chaining (CBC) mode, which produces a Message Integrity Code (MIC) or Message Authentication Code (MAC) appended to the transmitted data. The security modes supporting this are AES-CBC-MAC-32, AES-CBC-MAC-64 and AES-CBC-MAC-128, which differ on the size of the integrity code produced. This code is created with information from the 802.15.4 MAC header plus the payload data, and in such security modes the payload is transmitted unencrypted. Confidentiality, Data Authenticity and Integrity: The CTR and CBC modes may be jointly employed using the combined Counter with CBC-MAC AES/CCM encryption mode, which in IEEE 802.15.4 is used to support confidentiality as well as data authenticity and integrity for link-layer communications. This mode is supported in sensing platforms such as the TelosB in the CCM∗ variant, which also offers provides for integrity only and encryption-only security. This usage mode of AES provides confidentiality, message integrity and authenticity for data communications. The security modes are AES-CCM-32, AES-CCM-64 and AES-CCM-128, which again differ on the size of the MIC code following each message. AES-CCM modes require the transportation of all the security-related fields after the encrypted payload, as is illustrated in Fig.

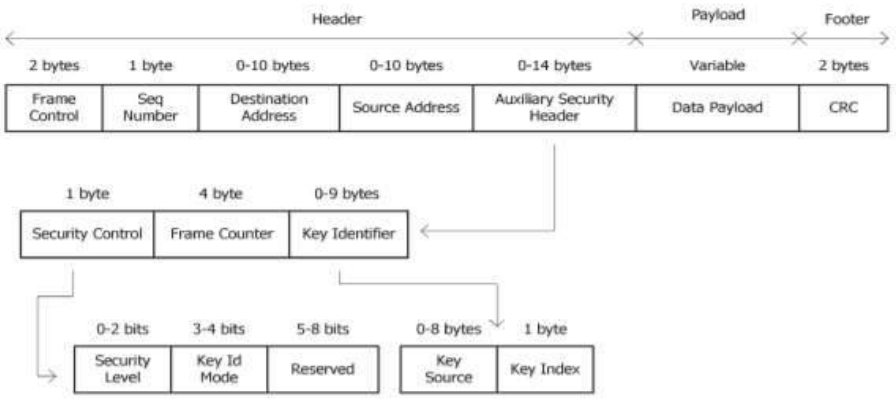


Fig 5.12 Security data and control fields in IEEE 802.15.4.

Security in 6LoWPAN No security mechanisms are currently defined in the context of the 6LoWPAN adaptation layer, but the relevant documents include discussions on the security vulnerabilities, requirements and approaches to consider for the usage of networklayer security, as we proceed to discuss. Later in Section VII we analyze research proposals on approaches to 6LoWPAN security, as well as the open research challenges and opportunities. Identification of Security Vulnerabilities: The discussion regarding security on RFC 4944 is related to the possibility of forging or accidentally duplicating EUI-64 interface addresses, which may consequently compromise the global uniqueness of 6LoWPAN interface identifiers. This document also discusses that Neighbor Discovery and mesh routing mechanisms on IEEE 802.15.4 environments may be susceptible to security threats, and that AES security at the link-layer may provide a basis for the development of mechanisms protecting against such threats, particularly for very constrained devices. Other interesting discussion is on the possibility of employing more powerful 6LoWPAN devices in order to support heavy securityrelated operations, also because such devices may support existing Internet security protocols, as such representing strategic places for the enforcement of security control mechanisms. The discussion concerning security on RFC 6282 focuses on the security issues posed by the usage of a mechanism inherited from RFC 4944, which enables the compression of a particular range of 16 UDP port numbers down to 4 bits. This document discusses that the overload of ports in this range, if employed with applications not honoring the reserved set for port compression, may increase the risk of an application getting the wrong type of payload or of an application misinterpreting the content of a message. As a result, RFC 6282 recommends that the usage of such ports be associated with a security mechanism employing MIC codes. Identification of Security Requirements and Strategies: The informational RFC 4919 discusses the addressing of security at various complementary protocol layers of the stack illustrated in Fig, considering that the most appropriate approach may depend on the application requirements and on the constraints of particular sensing devices. This document also identifies the possibility of employing security at the network layer using IPSec, together with the interest in investigating its applicability in the transport and tunnel usage modes. The discussion on security in RFC 6568 focuses on the possible approaches to adopt security in the light of the characteristics and constraints of wireless sensing devices. This document discusses threats due to the physical exposure of such devices, which may pose serious demands for its resiliency and survivability. It also discusses how IEEE 802.15.4 communications may facilitate attacks against the confidentiality, integrity, authenticity and availability of 6LoWPAN devices and communications. Rather than providing a specific approach to routing in 6LoWPAN environments, RFC 6606 provides guidelines that are useful in designing specific routing approaches. As with the previous standard documents, RFC 6606 identifies the importance of addressing security and the usefulness of AES/CCM available at the hardware of IEEE 802.15.4 sensing platforms. This document also discusses the importance of designing security mechanisms that are able to adapt to changes in the network topology and devices, rather than employing a static security configuration, given that many 6LoWPAN applications may employ networks that are dynamic in such respects. This document also discusses the importance of time

synchronization, self-organization and security localization in providing security for data and multi-hop routing control packets. Other important security requirements identified are the support of authenticated broadcasts and multicasts, and the verification of bidirectional links. RFC 6775 focuses on optimizations to enable Neighbor Discovery (ND) operations in 6LoWPAN environments, and also on the application of the threat model for ND operations defined in RFC 4861 to 6LoWPAN environments. Other possibilities discussed in this document consists in the adaptation of the SEcure Neighbor Discovery (SEND) and cryptographically generated addresses [36] mechanisms to 6LoWPAN environments.

Security in RPL The RPL specification defines secure versions of the various routing control messages, as well as three basic security modes. In Fig. illustrate the format of a secure RPL control message, transporting a Security field after the 4-byte ICMPv6 message header. The high order bit of the RPL Code field identifies whether or not security is applied to a given RPL message, which may thus be a secure DIS, DIO, DAO or DAOACK message. The format of the Security field is illustrated in Fig. 8. The information in the Security field indicates the level of security and the cryptographic algorithms employed to process security for the message. What this field doesn't include is the security—related data required to process security for the message, for example a Message Integrity Code (MIC) code or a signature. Instead, the security transformation itself states how the cryptographic fields should be employed in the context of the protected message. Support of Integrity and Data Authenticity: The current RPL specification [11] defines the employment of AES/CCM with 128-bit keys for MAC generation supporting integrity, and of RSA with SHA-256 for digital signatures supporting integrity and data authenticity. The LVL (Security Level) field indicates the provided packet security and allows for varying levels of data authentication and, optionally, of data confidentiality. RFC 6550 also defines various values to identify the presence of confidentiality, integrity and data authenticity with MAC-32 and MAC-64 authentication codes, as well as of 2048 and 3072- bit signatures using RSA. Support of Semantic Security and Protection Against Replay Attacks: A Consistency Check (CC) control message enables a sensing node to issue a challenge-response with the goal of validating another node's current counter value, for example in situations when a received message has an initialized (zero value) counter value and the receiver has an incoming counter currently maintained for the message originator. In this case the receiver initiates counter resynchronization by sending a CC message to the message source. Semantic security and protection against packet replay attacks is provided with the help of the Counter field, which may be used to transport a timestamp, as indicated by the T in Fig. The next byte in the Security section of the RPL control message identifies the security suite employed to provide security, while the Flags field is currently reserved. Support of Confidentiality: The secure variant of the various RPL control messages may also support confidentiality and delay protection. Regarding the employment of cryptographic algorithms in RPL, AES/CCM is adopted as the basis to support security in the current specification, while we note that other algorithms may be adopted in the future and appropriately identified in the Security section of a secure RPL control message. RPL control messages may be protected using both an integrated encryption and authentication suite, such as with AES/CCM, as well as schemes employing separate algorithms for encryption and authentication. The entire RPL message is within the scope of RPL security. MAC codes and signatures are calculated over the entire unsecured IPv6 packet, with the mutable fields of the packet zeroed. When a RPL ICMPv6 message is encrypted, encryption starts at the first byte after the Security section and continues to the last byte of the packet. The IPv6 header, the ICMPv6 header and the RPL message, up to the start of the Security field, are not encrypted, since those fields are required to correctly decrypt the packet. Support for Key Management: The KIM (Key Identifier Mode) field of the Security section illustrated in Fig. 8 indicates whether the cryptographic key required to process security for this message may be determined implicitly or explicitly. RFC 6550 currently defines different values for this field to thus supports different key management approaches, namely group keys, keys per pair of sensing devices, and digital signatures. This field supports various levels of granularity of packet protection, and is divided in a key source and key index subfields. The key source subfield

indicates the logical identifier of the originator of a group key, while the key index subfield, when present, allows unique identification of keys with the same originator.

Security Modes in RPL: As previously discussed, RPL defines how security is applied to routing control messages, and the current specification also defines the following three security modes:

- **Unsecured:** in this mode no security is applied to routing control messages, and this is the default usage mode of RPL.
- **Preinstalled:** this security mode may be employed by a device using a preconfigured symmetric key in order to join an existent RPL instance, either as a host or a router. This key is employed to support confidentiality, integrity and data authentication for routing control messages.

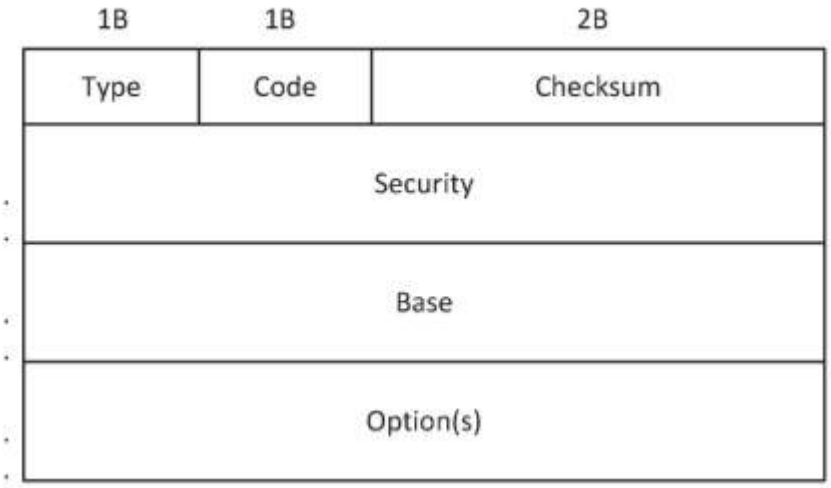


Fig 5.13 Secure RPL control message.

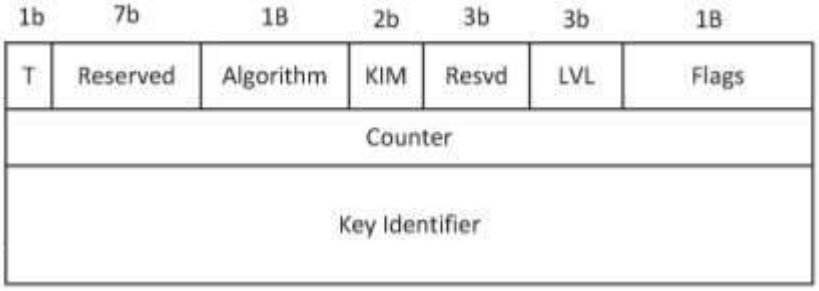


Fig 5.14 Security section of a secure RPL control message.

Authenticated: this security mode is appropriate for devices operating as routers. A device may initially join the network using a preconfigured key and the preinstalled security mode, and next obtain a different cryptographic key from a key authority with which it may start functioning as a router. The key authority is responsible for authenticating and authorizing the device for this purpose.

The RPL specification currently defines that the authenticated security mode must not be supported by symmetric cryptography, although it doesn't specify how asymmetric cryptography may be employed to support node authentication and key retrieval by the device intending to operate as a router. A more clear definition of such mechanisms is thus required, and future versions of the RPL standard may more clearly define how to support them. While not introducing additional security mechanisms, other documents relevant to RPL also include analysis on security aspects. This is the case of the informational RFC documents discussing routing requirements for the various application areas. Such documents discuss the importance of protecting

routing control messages with appropriate confidentiality, authentication and integrity. RFC 6551 specifies a set of link and node routing metrics appropriate to the characteristics and constraints of 6LoWPAN environments, and discusses the necessity of handling such metrics in a secure and trustful manner, including protection against nodes being able to falsify or lie in the advertisement of metrics, as a way to protect against attacks on routing operations.

SECURITY FOR IOT APPLICATION-LAYER COMMUNICATIONS

Application-layer communications are supported by the CoAP protocol, currently being designed by the Constrained RESTful Environments (CoRE) working group of the IETF. We next discuss the operation of the protocol as well as the mechanisms available to apply security to CoAP communications. A. Application-Layer Communications With CoAP The CoAP [12] protocol implements a set of techniques to compress application-layer protocol metadata without compromising application inter-operability, in conformance with the representational state transfer (REST) architecture of the web. CoAP is currently defined only for UDP communications over 6LoWPAN, although the adoption of transport-layer approaches with characteristics more close to protocols such as the Transmission Control Protocol (TCP) is still open to debate, with ongoing research addressing the adaptation of TCP for 6LoWPAN environments. Application-layer communications may enable IoT sensing applications to interoperate with existing Internet applications without requiring specialized application oriented code or translation mechanisms. CoAP restricts the HTTP dialect to a subset that is well suited to the constraints of 6LoWPAN sensing devices, and may enable abstracted communications Fig. 9. Format of a CoAP message header. between users, applications and such devices, in the context of IoT applications. The CoAP protocol provides a request and response communications model between application endpoints and enables the usage of key concepts of the web, namely the usage of URI addresses to identify the resources available on constrained sensing devices. The protocol may support end to-end communications at the application-layer between constrained IoT sensing devices and other Internet entities, using only CoAP or in alternative by translating HTTP to CoAP at a reverse or forward gateway. Messages in the CoAP protocol are exchange asynchronously between two endpoints, and used to transport CoAP requests and responses. Since such messages are transported over unreliable UDP communications, CoAP provides a lightweight reliability mechanism. Using this mechanism CoAP messages may be marked as Confirmable, for which the sender activates a simple stop-and-wait retransmission mechanism with exponential backoff. The receiver must acknowledge a Confirmable message with a corresponding Acknowledge message or, if it lacks context to process the message properly, reject it with a Reset message. Acknowledge or Reset messages are related to a Confirmable message by means of a Message ID, along with the address of the corresponding endpoint. CoAP messages may also be transmitted less reliably if marked as Non-Confirmable, in which case the recipient does not acknowledge the message. Similarly to HTTP, CoAP defines a set of method and response codes available to applications. Other than a basic set of information, most of the information in CoAP is transported using options. Options defined for the CoAP Protocol may be critical, elective, safe or unsafe. A critical option is one that an endpoint must understand, while an elective option may be ignored by an endpoint not recognizing it. Safe and unsafe options determine how an option may be processed by an intermediary entity. An unsafe option needs to be understood by the proxy in order to be forwarded, while a safe option may be forwarded even if the proxy is unable to process it. The CoAP header and message format is illustrated in Fig. The message starts with a 4-byte fixed header, formed by the Version field (2 bits), the T (message type) field (2 bits), the TKL (Token Length) field (4 bits), the Code field (8 bits) and the Message ID (16 bits). The token in practice enables a CoAP entity to perform matching of CoAP requests and replies, while the message ID supports duplicate detection and optional reliability. The options adopted in CoAP are defined in the Type-lengthvalue (TLV) format, by specifying its option number followed by its length and value. CoAP currently defines the Uri-Host, Uri-Port, Uri-Path and Uri-Query options enabling the identification of the target resource of a request, Content-Format to specify the representation format of the message payload, and Max-Age to indicate the maximum time a CoAP response may be cached before being considered not fresh, among others. Regarding

security, rather than designing mechanisms to support (object) security directly in the context of application layer communications, CoAP adopts DTLS at the transport layer to transparently apply security to all CoAP messages in a given communications session. The protocol also defines four security modes.

Security in CoAP The CoAP Protocol defines bindings to DTLS (Datagram Transport-Layer Security) [45] to secure CoAP messages, along with a few mandatory minimal configurations appropriate for constrained environments. Support for Confidentiality, Authentication, Integrity, NonRepudiation and Protection Against Replay Attacks: The adoption of DTLS implies that security is supported at the transport-layer, rather than being designed in the context of the application-layer protocol. DTLS provides guarantees in terms of confidentiality, integrity, authentication and non-repudiation for application-layer communications using CoAP. DTLS is in practice TLS with added features to deal with the unreliable nature of UDP communications. Fig.illustrates the availability of payload space for applications in IEEE 802.15.4 and 6LoPWAN communication environments in the presence of CoAP and DTLS. Once the initial DTLS handshake is completed, DTLS adds a limited per-datagram overhead of 13 bytes, not counting any initialization vectors, integrity check values or the padding that may be required by the cipher suite employed. As considered in Fig. 10, shared-context 6LoWPAN header compression requires 10 bytes for an UDP/IPv6 header, while the CoAP fixed header requires 4 bytes. The impact of DTLS on constrained wireless sensing devices is due to the cost of supporting the initial handshake plus the processing of security for each exchanged CoAP messages. The impact of DTLS on constrained wireless sensing devices is due to the cost of supporting the initial handshake plus the processing of security for each exchanged CoAP messages. Similarly to other approaches to security in 6LoWPAN environments, AES/CCM is adopted as the cryptographic algorithm to support fundamental security requirements in the current CoAP specification. Security against replay attacks may also be achieved in the context of DTLS, using a different nonce value for each secured CoAP packet. Security Modes in CoAP: In addition to the adoption of DTLS, CoAP currently defines four security modes that applications may employ. Those security modes essentially differ on how authentication and key negotiation is performed, as follows: • NoSec: this mode in practice provides no security, and CoAP messages are transmitted without security applied. • PreSharedKey: this security mode may be employed by sensing devices that are pre-programmed with the symmetric cryptographic keys required to support secure communications with other devices or groups of devices. This mode may be appropriate to applications employing devices that are unable to support public-key cryptography, or for which it is convenient to employ security preconfiguration. Applications may use one key per destination device or in alternative a single key for a group of destination devices. • RawPublicKey: this security mode is appropriate for devices requiring authentication based on public keys, but which are unable to participate in public-key infrastructures. A given device must be preprogrammed with an asymmetric key pair that may be validated using an outof-band mechanism and possibly programmed as part of the manufacturing process, while without a certificate. The device has an identity calculated from its public key and a list of identities and public keys of the nodes it can communicate with. This security mode is defined as mandatory to implement in CoAP. • Certificates: this security mode also supports authentication based on public-keys, but for applications that are able to participate in a certification chain for certificate validation purposes. This security mode thus assumes the availability and usage of a security infrastructure. The device has an asymmetric key pair with an X.509 certificate that binds it to its Authority Name and is signed by some common trusted root. The device also has a list of root trust anchors that can be used for certificate validation.

An important aspect of CoAP security using DTLS is that Elliptic Curve Cryptography (ECC) [48] is adopted to support the RawPublicKey and Certificates security modes. ECC supports device authentication using the Elliptic Curve Digital Signature Algorithm (ECDSA), and also key agreement using the ECC Diffie-Hellman counterpart, the Elliptic Curve Diffie-Hellman Algorithm with Ephemeral keys (ECDHE). The NoSec security mode corresponds to a device sending packets without security, using the “coap” scheme in URI addresses identifying resources available on CoAP servers. On the other end, accesses to resources with DTLS use the “coaps” scheme, and in this case a security association at the transport-layer using DTLS must exist between

the CoAP client and the CoAP server.

The current CoAP specification defines a mandatory-toimplement cipher suite for each security mode, based on the usage of AES/CCM and ECC cryptographic operations, as follows:

• Applications supporting the PreSharedKey security mode are required to support at least the TLS_PSK_WITH_AES_ 128_CCM_8 suite, which supports authentication using pre-shared symmetric keys and 8-byte nonce values, and encrypts and produces 8-byte integrity codes.

• Applications supporting the RawPublicKey CoAP security mode are required to support the TLS_ECDHE_ ECDSA_WITH_AES_128_CCM_8 security suite using ECDSA-capable public keys. This security mode also employs SHA-256 to compute hashes.

• Applications supporting the Certificates security mode are also required to support the TLS_ECDHE_ECDSA_ WITH_AES_128_CCM_8 cipher suite. Regarding the usage of public-keys transported in X.509 certificates, the SubjectPublicKeyInfo field in a X.509 certificate defines how the corresponding public key must be employed for ECC computations. The certificate must also contain a signature created using ECDSA and SHA-256. Applications using devices with a shared key plus a certificate must also support TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA.

In addition to the cipher suites it may be expected that further security suites may be adopted in future versions of CoAP, as this would enable a better adaptation of the various security modes to different applications and types of sensing platforms. CoAP also doesn't currently define or adopt any solution to address key management, other than the assumption that initial keys are available resulting from the DTLS authentication handshake.

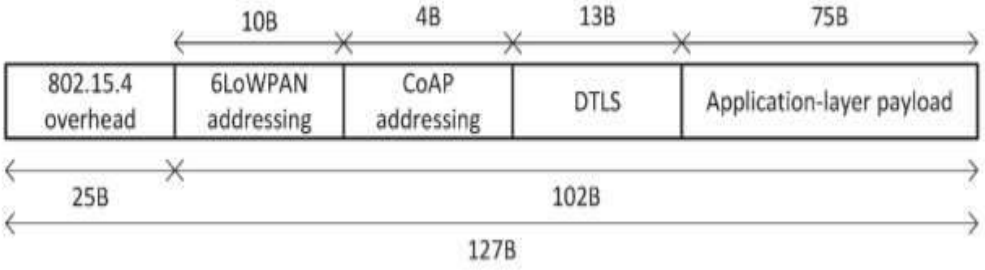


Fig : 5.15 Payload space with DTLS on 6LoWPAN environments.

TEXT / REFERENCE BOOKS

1 Jan Holler, VlasiosTsiatsis, Catherine Mulligan, Stefan Avesand, Stamatis Karnouskos, David Boyle, From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence Press, 2014.

2 Cirani, Simone, Gianluigi Ferrari, Marco Picone, and Luca Veltri. Internet of Things: Architectures, Protocols and Standards. John Wiley & Sons, 2018.

3 Hassan, Qusay F., ed. Internet of Things A to Z: technologies and applications. John Wiley & Sons, 2018.

4 Holler, Jan, Vlasios Tsiatsis, Catherine Mulligan, Stamatis Karnouskos, Stefan Avesand, and David Boyle. Internet of Things. Academic Press, 2014.

5 Hersent, Olivier, David Boswarthick, and Omar Elloumi. The internet of things: Key applications and protocols. John Wiley & Sons, 2011.

6 Bernd Scholz- -3-642-19156-5 e-ISBN Architecting the Internet of Things, ISBN 978-3-642-19156-5, 978-3-642-19157-2, Springer