INTERNET OF THINGS TECHNOLOGY
[As per Choice Based Credit System (CBCS) scheme]
(Effective from the academic year 2016 -2017)
SEMESTER – VIII

Subject Code	15CS81	IA Marks	20
Number of Lecture Hours/Week	04	Exam Marks	80
Total Number of Lecture Hours	50	Exam Hours	03

CREDITS – 04

HKBK COLLEGE OF ENGINEERING, BENGALURU

Course Objectives: This course will enable students to
- Assess the genesis and impact of IoT applications, architectures in real world.   - Illustrate diverse methods of deploying smart objects and connect them to network.   - Compare different Application protocols for IoT.   - Infer the role of Data Analytics and Security in IoT.   - Identifysensor technologies for sensing real world entities and understand the role of IoT in various domains of Industry.

Course Outcomes: After studying this course, students will be able to
- Interpret the impact and challenges posed by IoT networks leading to new architectural models.   - Compare and contrast the deployment of smart objects and the technologies to connect them to network.   - Appraise the role of IoT protocols for efficient network communication.   - Elaborate the need for Data Analytics and Security in IoT.   - Illustrate different sensor technologies for sensing real world entities and identify the applications of IoT in Industry.

Syllabus- Internet Of Things Technology

Module – 1	Teaching Hours
What is IoT, Genesis of IoT, IoT and Digitization, IoT Impact, Convergence of IT and IoT, IoT Challenges, IoT Network Architecture and Design, Drivers Behind New Network Architectures, Comparing IoT Architectures, A Simplified IoT Architecture, The Core IoT Functional Stack, IoT Data Management and Compute Stack.	10 Hours
Module – 2	
Smart Objects: The "Things" in IoT, Sensors, Actuators, and Smart Objects, Sensor Networks, Connecting Smart Objects, Communications Criteria, IoT Access Technologies.	10 Hours
Module – 3	
IP as the IoT Network Layer, The Business Case for IP, The need for Optimization, Optimizing IP for IoT, Profiles and Compliances, Application Protocols for IoT, The Transport Layer, IoT Application Transport Methods.	10 Hours

Syllabus- Internet Of Things Technology

Module – 4	
Data and Analytics for IoT, An Introduction to Data Analytics for IoT, Machine Learning, Big Data Analytics Tools and Technology, Edge Streaming Analytics, Network Analytics, Securing IoT, A Brief History of OT Security, Common Challenges in OT Security, How IT and OT Security Practices and Systems Vary, Formal Risk Analysis Structures: OCTAVE and FAIR, The Phased Application of Security in an Operational Environment	**10 Hours**
Module – 5	
IoT Physical Devices and Endpoints - Arduino UNO: Introduction to Arduino, Arduino UNO, Installing the Software, Fundamentals of Arduino Programming. IoT Physical Devices and Endpoints - RaspberryPi: Introduction to RaspberryPi, About the RaspberryPi Board: Hardware Layout, Operating Systems on RaspberryPi, Configuring RaspberryPi, Programming RaspberryPi with Python, Wireless Temperature Monitoring System Using Pi, DS18B20 Temperature Sensor, Connecting Raspberry Pi via SSH, Accessing Temperature from DS18B20 sensors, Remote access to RaspberryPi, Smart and Connected Cities, An IoT Strategy for Smarter Cities, Smart City IoT Architecture, Smart City Security Architecture, Smart City Use-Case Examples.	**10 Hours**

Question paper pattern:

The question paper will have ten questions.

There will be 2 questions from each module.

Each question will have questions covering all the topics under a module.

The students will have to answer 5 full questions, selecting one full question from each module.

Text Books:

1. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Robert Barton, Jerome Henry,**"IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things"**, 1^{st}Edition, Pearson Education (Cisco Press Indian Reprint). (**ISBN:** 978-9386873743)

2. Srinivasa K G, **"Internet of Things",**CENGAGE Leaning India, 2017

Reference Books:

1. Vijay Madisetti and ArshdeepBahga, **"Internet of Things (A Hands-on-Approach)"**, 1^{st}Edition, VPT, 2014. (**ISBN:** 978-8173719547)

2. Raj Kamal, **"Internet of Things: Architecture and Design Principles"**, 1st Edition, McGraw Hill Education, 2017. (**ISBN:** 978-9352605224)

Module – 3 IP as the IoT Network Layer

IP as the IoT Network Layer:

➢ **The Business Case for IP**: This section discusses the **advantages of IP** from an IoT perspective and introduces the concepts of adoption and adaptation.

➢ **The Need for Optimization**: This section dives into **the challenges of constrained nodes** and devices when deploying IP. This section also looks at the **migration from IPv4 to IPv6** and how it affects IoT networks.

➢ **Optimizing IP for IoT**: This section **explores the common protocols** and technologies in IoT networks utilizing IP, including 6LoWPAN, 6TiSCH, and RPL.

➢ **Profiles and Compliances**: This section provides a **summary** of some of the most significant **organizations and standards** bodies involved with IP connectivity and IoT.

Module – 3 IP as the IoT Network Layer

IP as the IoT Network Layer:

➢ **The Business Case for IP**:

➢ **Data flowing from or to "things" is consumed, controlled, or monitored** by data center servers either in the cloud or in locations that may be distributed or centralized.

➢ Dedicated applications are then run over **virtualized or traditional** operating systems or on network edge platforms (for example, fog computing).

The Business Case for IP:

➤ **lightweight applications communicate with the data center servers.**

➤ **the system solutions combining various physical and data link layers** call **for an architectural approach with a common layer(s) independent from the lower (connectivity) and/or upper (application) layers.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

➢ **One of the main differences between traditional information technology (IT) and operational technology (OT) is the lifetime of the underlying technologies and products.**

➢ **IP is able to maintain its operations for large numbers of devices and users, such as the 3 billion Internet users.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➢ **1. Open and standards-based: The Internet of Things creates a new paradigm in which devices, applications, and users can leverage a large set of devices and functionalities while guaranteeing interchangeability and interoperability, security, and management.**

➢ **This calls for implementation, validation, and deployment of open, standards-based solutions.**

➢ **While many standards development organizations (SDOs) are working on Internet of Things definitions, frameworks, applications, and technologies, the role of the Internet Engineering Task Force (IETF) as the foundation for specifying and optimizing the network and transport layers.**

➢ **The IETF is an open standards body that focuses on the development of the Internet Protocol suite and related Internet technologies and protocols.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

- ➢ **2. Versatile: A large spectrum of access technologies is available to offer connectivity of "things" in the last mile.**

- ➢ **Additional protocols and technologies are also used to transport IoT data through backhaul links and in the data center.**

- ➢ **So, the layered IP architecture is well equipped to cope with any type of physical and data link layers.**

- ➢ **This makes IP ideal as a long term investment because various protocols at these layers can be used in a deployment now and over time, without requiring changes to the whole solution architecture and data flow.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➤ **3. Ubiquitous:** All recent operating system releases, from general-purpose computers and servers to lightweight embedded systems (TinyOS, Contiki, and so on), have **an integrated dual (IPv4 and IPv6) IP stack** that gets enhanced over time.

➤ In addition, IoT application protocols in many industrial OT solutions have been **updated in recent years to run over IP.**

➤ recent standardization efforts in several areas are adding **IPv6.**

➤ IP is the most **pervasive protocol** when supported across the various IoT solutions and industry verticals.

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➢ **4. Scalable: As** the common protocol of the Internet, IP has been massively deployed and tested for robust scalability.

➢ **Millions of private and public IP infrastructure nodes have been operational for years, offering strong foundations for those not familiar with IP network management.**

➢ **adding huge numbers of "things" to private and public infrastructures may require optimizations and design rules specific to the new devices.**

➢ **recent evolution -voice and video endpoints are also integrated over IP.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

- **5. Manageable and highly secure:** Communications infrastructure **requires** appropriate management and **security capabilities** for proper operations.

- One of the benefits that comes from 30 years of operational IP networks is the well-understood **network management and security protocols, mechanisms, and toolsets** that are widely available.

- Adopting IP network management also brings an **operational business application** to OT.

- Well-known **network and security management** tools are easily **leveraged** with an IP network layer.

- industry is challenged in **securing constrained nodes**, **handling legacy OT protocols**, and scaling operations.

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➢ **6. Stable and resilient:** IP has been around for **30 ye**ars, and it is clear that **IP is a workable** solution.

➢ IP has a large and well-established knowledge base and, more importantly, it has been used for years in **critical infrastructures**, such as **financial and defense** networks.

➢ In addition, IP has been **deployed** for critical services, such as **voice and video**, which have already transitioned from closed environments to open IP standards.

➢ Finally, its **stability and resiliency** benefit from the large ecosystem of IT professionals who can help **design, deploy, and operate** IP-based solutions.

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➢ **7. Consumers' market adoption: When developing IoT solutions and products targeting the consumer market, vendors know that consumers' access to applications and devices will occur predominantly over broadband and mobile wireless infrastructure.**

➢ **The main consumer devices range from smart phones to tablets and PCs.**

➢ **The common protocol that links IoT in the consumer space to these devices is IP.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

➤ **8. The innovation factor: The past two decades have largely established the adoption of IP as a factor for increased innovation.**

➤ **IP is the underlying protocol for applications ranging from file transfer and e-mail to the World Wide Web, e-commerce, social networking, mobility, and more.**

➤ **Even the recent computing evolution from PC to mobile and mainframes to cloud services are perfect demonstrations of the innovative ground enabled by IP.**

➤ **Innovations in IoT can also leverage an IP underpinning.**

Module – 3 IP as the IoT Network Layer

The Key Advantages of Internet Protocol:

Quick review of the key advantages of the IP suite for the Internet of Things:

Summary:

➢ adoption of IP provides a **solid foundation** for the Internet of Things by allowing **secured and manageable** bidirectional data communication capabilities between all devices in a network.

➢ IP is a **standards-based protocol** that is **ubiquitous, scalable, versatile, and stable**.

➢ Network services such as **naming, time distribution, traffic prioritization, isolation**, and so on are well known and developed techniques that can be leveraged with IP.

Module – 3 IP as the IoT Network Layer

Adoption or Adaptation of the Internet Protocol:

1. Adaptation means application layered gateways (ALGs) must be implemented to ensure the translation between non-IP and IP layers.

2. Adoption involves replacing all non-IP layers with their IP layer counterparts, simplifying the deployment model and operations.

Module – 3 IP as the IoT Network Layer

Adoption or Adaptation of the Internet Protocol:

Supervisory control and data acquisition (SCADA) applications are typical examples of vertical market deployments that **operate** both the IP adaptation model and the adoption model.

➢ SCADA is an **automation control system** for remote monitoring and control of equipment.

➢ Implementations that make use of IP adaptation have SCADA devices attached through **serial interfaces to a gateway tunneling or translating** the traffic.

➢ With the IP adoption model, SCADA devices are attached via **Ethernet to switches and routers forwarding their IPv4 traffic**.

Adoption or Adaptation of the Internet Protocol:

Another example is a ZigBee solution that runs a non-IP stack between devices and a ZigBee gateway that forwards traffic to an application server.

A ZigBee gateway often acts as a translator between the ZigBee and IP protocol stacks.

Module – 3 IP as the IoT Network Layer

Adoption or Adaptation of the Internet Protocol:

consider the following factors when trying to determine which model is best suited for last-mile connectivity:

1. **Bidirectional versus unidirectional data flow:**

➢ While bidirectional communications are generally expected, some last-mile technologies offer **optimization for unidirectional communication.**

➢ For example, different classes of IoT devices, as defined in RFC 7228, may only **infrequently** need to report a few bytes of data to an application.

➢ These sorts of devices, particularly ones that communicate through **LPWA technologies**, include fire alarms sending alerts or daily test reports, electrical switches being pushed on or off, and water or gas meters **sending weekly indexes**.

➢ for example, if there is only **one-way communication** to upload data to an application, then it is **not possible** to download new software or firmware to the devices. This makes integrating **new features and bug and security fixes** more difficult.

Module – 3 IP as the IoT Network Layer

2. Overhead for last-mile communications paths:

- ➢   IP adoption implies a layered architecture with a **per-packet overhead** that varies depending on the IP version.

- ➢   IPv4 has **20 bytes of header** at a minimum, and IPv6 has **40 bytes** at the IP network layer.

- ➢   For the IP transport layer, UDP has **8 bytes of header overhead**, while TCP has a minimum of **20 bytes**.

- ➢   If the data to be forwarded by a device is **infrequent** and only a few bytes, we can potentially have more **header overhead** than device data—again, particularly in the case of LPWA technologies.

- ➢   Consequently, decide whether the **IP adoption model is necessary** and how it can be optimized.

- ➢   This consideration applies to control plane traffic that is run over **IP for low-bandwidth**, last-mile links.

- ➢   **Routing protocol** and other verbose network services may either **not be required** or call for optimization.

Module – 3 IP as the IoT Network Layer

3. Data flow model:

➢ **One benefit of the IP adoption model is the end-to-end nature of communications.**

➢ **Any node can easily exchange data with any other node in a network, although security, privacy, and other factors may put controls and limits on the "end-to-end" concept.**

➢ **However, in many IoT solutions, a device's data flow is limited to one or two applications.**

➢ **In this case, the adaptation model can work because translation of traffic needs to occur only between the end device and one or two application servers.**

➢ **Depending on the network topology and the data flow needed, both IP adaptation and adoption models have roles to play in last-mile connectivity.**

Module – 3 IP as the IoT Network Layer

4. Network diversity:

➢ **One of the drawbacks of the adaptation model is a general dependency on single PHY and MAC layers.**

➢ **For example, ZigBee devices must only be deployed in ZigBee network islands.**

➢ **This same restriction holds for ITU G.9903 G3-PLC nodes.**

➢ **Therefore, a deployment must consider which applications have to run on the gateway connecting these islands and the rest of the world.**

➢ **Integration and coexistence of new physical and MAC layers or new applications impact how deployment and operations have to be planned.**

Module – 3 IP as the IoT Network Layer

The Need for Optimization:

Optimizations are needed at various layers of the IP stack to handle the restrictions that are present in IoT networks.

Constrained Nodes:

In IoT solutions, different classes of devices coexist. Depending on its functions in a network, a "thing" architecture may or may not offer similar characteristics compared to a generic PC or server in an IT environment.

Another limit is that this network protocol stack on an IoT node may be required to communicate through an unreliable path.

Even if a full IP stack is available on the node, this causes problems such as limited or unpredictable throughput and low convergence when a topology change occurs.

Module – 3 IP as the IoT Network Layer

Constrained Nodes:

Finally, **power consumption** is a key characteristic of constrained nodes.

Many IoT devices are **battery powered**, with lifetime battery requirements varying from a **few months** to **10+ years**.

This drives the selection of networking technologies since high-speed ones, such as **Ethernet, Wi-Fi, and cellular**, are not (yet) capable of **multi-year battery life**.

Current capabilities practically allow **less than a year** for these technologies on battery-powered nodes.

power consumption is much less of a concern on nodes that **do not require** batteries as an energy source.

Module – 3 IP as the IoT Network Layer

IoT constrained nodes can be classified as follows:

1. **Devices that are very constrained in resources, may communicate infrequently to transmit a few bytes, and may have limited security and management capabilities:**

   - **This drives the need for the IP adaptation model, where nodes communicate through gateways and proxies.**

2. **Devices with enough power and capacities to implement a stripped downIP stack or non-IP stack:**

   - **implement either an optimized IP stack and directly communicate with application servers (adoption model) or go for an IP or non-IP stack and communicate through gateways and proxies (adaptation model).**

Module – 3 IP as the IoT Network Layer

IoT constrained nodes can be classified as follows:

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

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

- The costs of **computing power, memory, storage resources, and power consumption** are generally decreasing.

- At the same time, networking technologies continue **to improve** and offer **more bandwidth** and **reliability**.

- In the future, the push to optimize IP for constrained nodes will lessen as technology improvements and cost decreases address many of these challenges.

Constrained Networks:

- **In the early years of the Internet, network bandwidth capacity was restrained due to technical limitations.**

- **Connections often depended on low-speed modems for transferring data.**

- **However, these low-speed connections demonstrated that IP could run over low-bandwidth networks.**

- **high-speed connections are not usable by some IoT devices in the last mile.**

- **The reasons include the implementation of technologies with low bandwidth, limited distance and bandwidth due to regulated transmit power, and lack of or limited network services.**

Module – 3 IP as the IoT Network Layer

Constrained Networks:

- A constrained network can have **high latency and a high potential** for packet loss.

- Constrained networks have **unique characteristics** and **requirements.**

- In contrast with typical IP networks, where **highly stable and fast links** are available, constrained networks are **limited by low-power, low-bandwidth links** (wireless and wired).

- They **operate** between a **few kbps** and a few **hundred kbps** and may utilize a star, mesh, or combined network topologies, ensuring proper operations.

- **packet delivery rate (PDR**) to oscillate between low and high percentages.

- Large bursts of unpredictable errors and even **loss of connectivity** at times may occur.

Constrained Networks:

- **Due to the low bandwidth, a constrained network that overreacts can lead to a network collapse.**

- **In summary, constrained nodes and networks pose major challenges for IoT connectivity in the last mile.**

- **This in turn has led various standards organizations to work on optimizing protocols for IoT**

Module – 3 IP as the IoT Network Layer

IP Versions:

- **For 20+ years, the IETF has been working on transitioning the Internet from IP version 4 to IP version 6.**

- **The main driving force has been the lack of address space in IPv4 as the Internet has grown.**

- **IPv6 has a much larger range of addresses that should not be exhausted for the foreseeable future.**

- **Today, both versions of IP run over the Internet, but most traffic is still IPv4 based.**

Module – 3 IP as the IoT Network Layer

IP Versions:

- **A variety of factors dictate whether IPv4, IPv6, or both can be used in an IoT solution.**

- **Most often these factors include a legacy protocol or technology that supports only IPv4.**

The following are some of the main factors applicable to IPv4 and IPv6 support in an IoT solution:

1. **Application Protocol:**

   - **IoT devices implementing Ethernet or Wi-Fi interfaces can communicate over both IPv4 and IPv6, but the application protocol may dictate the choice of the IP version.**

   - **For example, SCADA protocols such as DNP3/IP (IEEE 1815), Modbus TCP, or the IEC 60870-5-104 standards are specified only for IPv4.**

Module – 3 IP as the IoT Network Layer

IP Versions:

2. Cellular Provider and Technology:

- **IoT devices with cellular modems** are dependent on the generation of the cellular technology as well as the **data services** offered by the provider.

- For the first three generations of data **services—GPRS, Edge, and 3G—IPv4** is the base protocol version.

- Consequently, if IPv6 is used with these generations, it must be **tunneled** over IPv4.

- On **4G/LTE** networks, data services can use **IPv4 or IPv6** as a base protocol, depending on the provider.

Module – 3 IP as the IoT Network Layer

IP Versions:

3. Serial Communications:

➢ **Many legacy devices in certain industries, such as manufacturing and utilities, communicate through serial lines Data is transferred using either proprietary or standards-based protocols, such as DNP3, Modbus, or IEC 60870-5-101.**

➢ **In the past, communicating this serial data over any sort of distance could be handled by an analog modem connection.**

➢ **connect the serial port of the legacy device to a nearby serial port on a piece of communications equipment, typically a router. This local router then forwards the serial traffic over IP to the central server for processing.**

➢ **Encapsulation of serial protocols over IP leverages mechanisms such as raw socket TCP or UDP.**

➢ **While raw socket sessions can run over both IPv4 and IPv6, current implementations are mostly available for IPv4 only.**

Module – 3 IP as the IoT Network Layer

IP Versions:

4. IPv6 Adaptation Layer:

➢ **IPv6-only adaptation layers for some physical and data link layers for recently standardized IoT protocols support only IPv6.**

➢ **While the most common physical and data link layers (Ethernet, Wi-Fi, and so on) stipulate adaptation layers for both versions, newer technologies, such as IEEE 802.15.4 (Wireless Personal Area Network), IEEE 1901.2, and ITU G.9903 (Narrowband Power Line Communications) only have an IPv6 adaptation layer specified.**

➢ **This means that any device implementing a technology that requires an IPv6 adaptation layer must communicate over an IPv6-only subnetwork.**

➢ **This is reinforced by the IETF routing protocol for LLNs, RPL, which is IPv6 only.**

Optimizing IP for IoT:

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

➢ **The following sections introduce some of these optimizations already available from the market or under development by the IETF.**

➢ **Next Figure highlights the TCP/IP layers where optimization is applied.**

Optimizing IP for IoT:
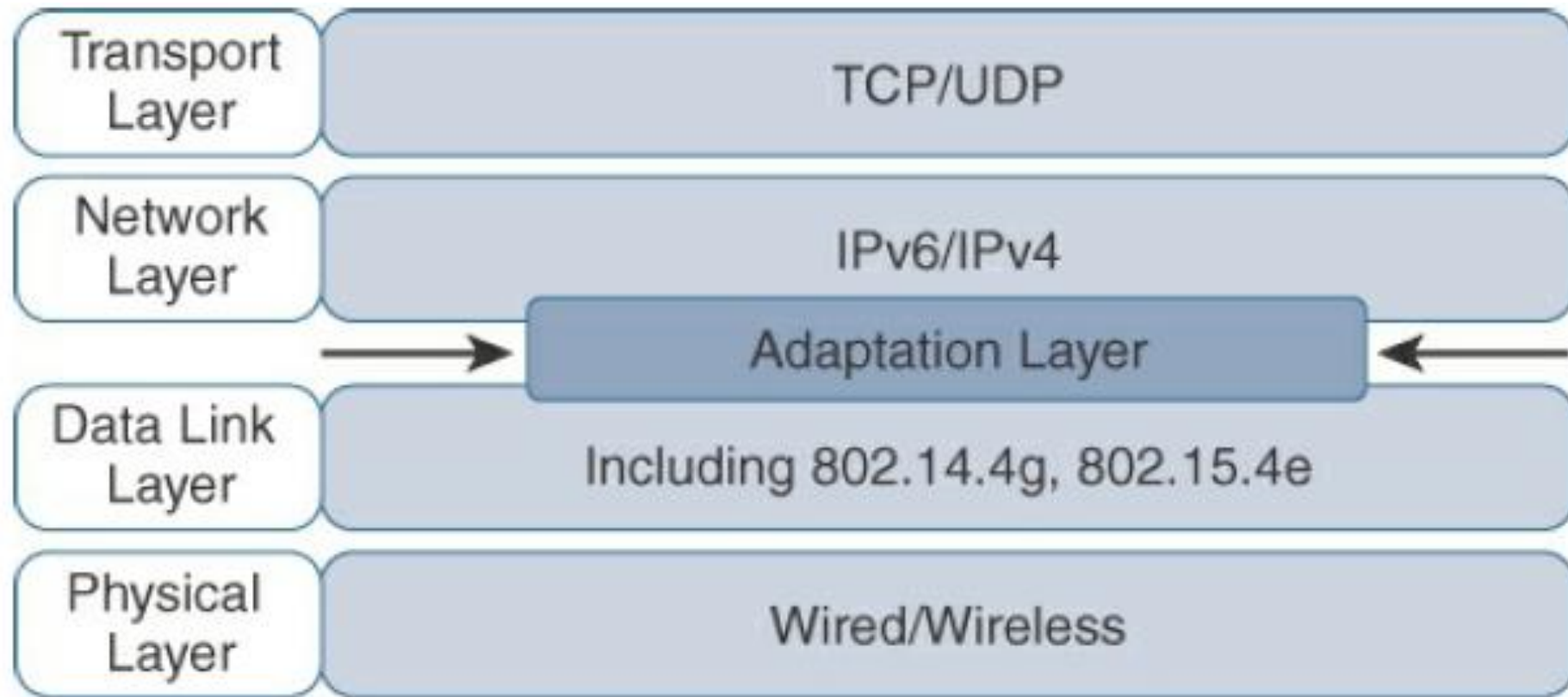
Optimizing IP for IoT Using an Adaptation Layer

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

➢ In the IP architecture, the **transport of IP packets** over any given **Layer 1 (PHY) and Layer 2 (MAC) protocol** must be defined and documented.

➢ The model for **packaging IP** into **lower-layer protocols** is often referred to as **an adaptation layer.**

➢ Unless the technology is proprietary, IP adaptation layers are typically defined by an IETF working group and released as a **Request for Comments (RFC).**

➢ An RFC is a **publication from the IETF** that officially **documents Internet standards, specifications, protocols, procedures, and events.**

➢ For example, RFC 864 **describes** how an IPv4 packet gets **encapsulated over an Ethernet frame**, and RFC **2464** describes how the same function is performed for an **IPv6 packet.**

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

➢ **The main examples of adaptation layers optimized for constrained nodes or "things" are the ones under the 6LoWPAN working group and its successor, the 6Lo working group.**

➢ **The initial focus of the 6LoWPAN working group was to optimize the transmission of IPv6 packets over constrained networks such as IEEE 802.15.4.**

➢ **Figure shows an example of an IoT protocol stack using the 6LoWPAN adaptation layer beside the well-known IP protocol stack for reference.**
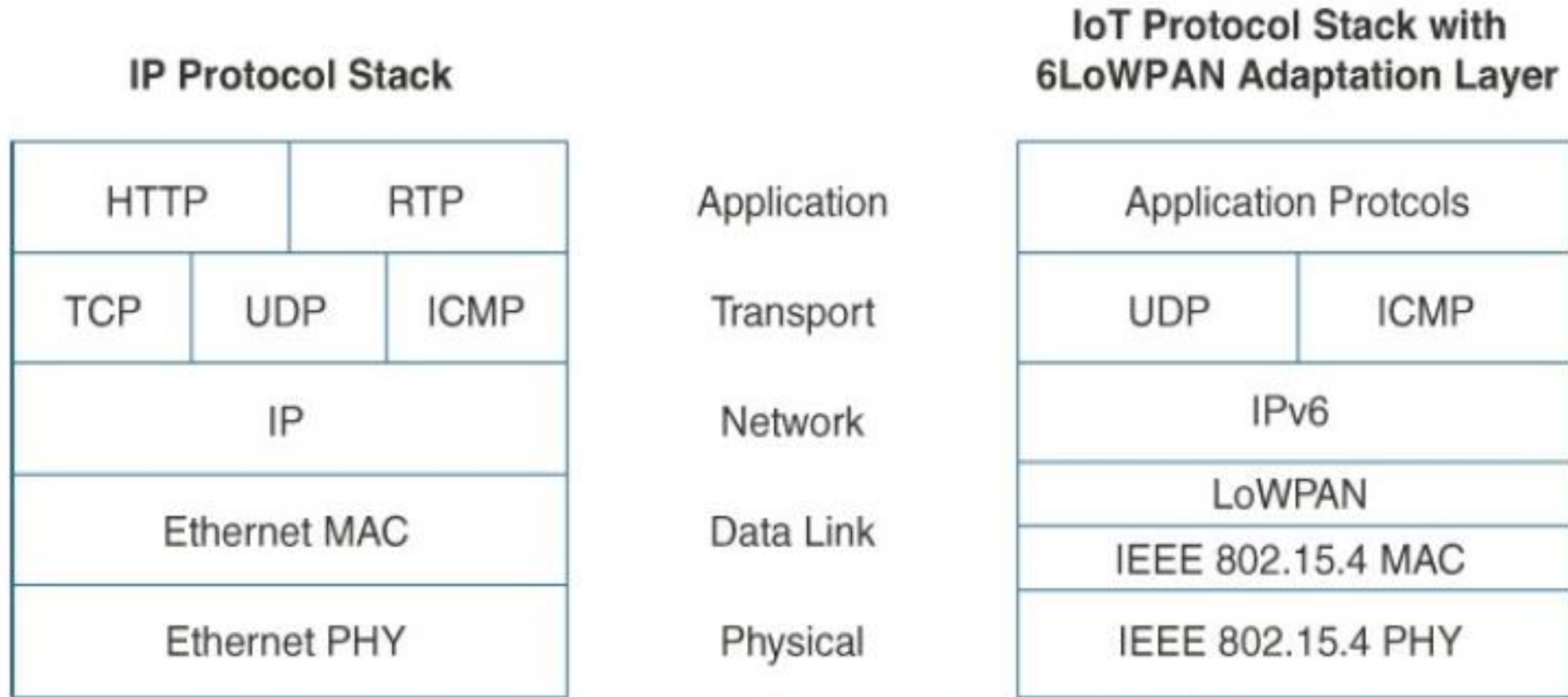
From 6LoWPAN to 6Lo:

Comparison of an IoT Protocol Stack Utilizing 6LoWPAN and an IP Protocol Stack

42

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

➢ The 6LoWPAN working group **published** several **RFCs**, but **RFC 4994** is foundational because it **defines frame headers** for the capabilities of **header compression, fragmentation, and mesh addressing.**

➢ These headers can be stacked in the **adaptation layer** to keep these concepts separate while **enforcing a structured method** for expressing each capability.

➢ Depending on the implementation, **all, none, or any combination** of these capabilities and their **corresponding headers** can be enabled.

➢ Figure shows some examples of **typical 6LoWPAN header stacks**.

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

6LoWPAN Header Stacks
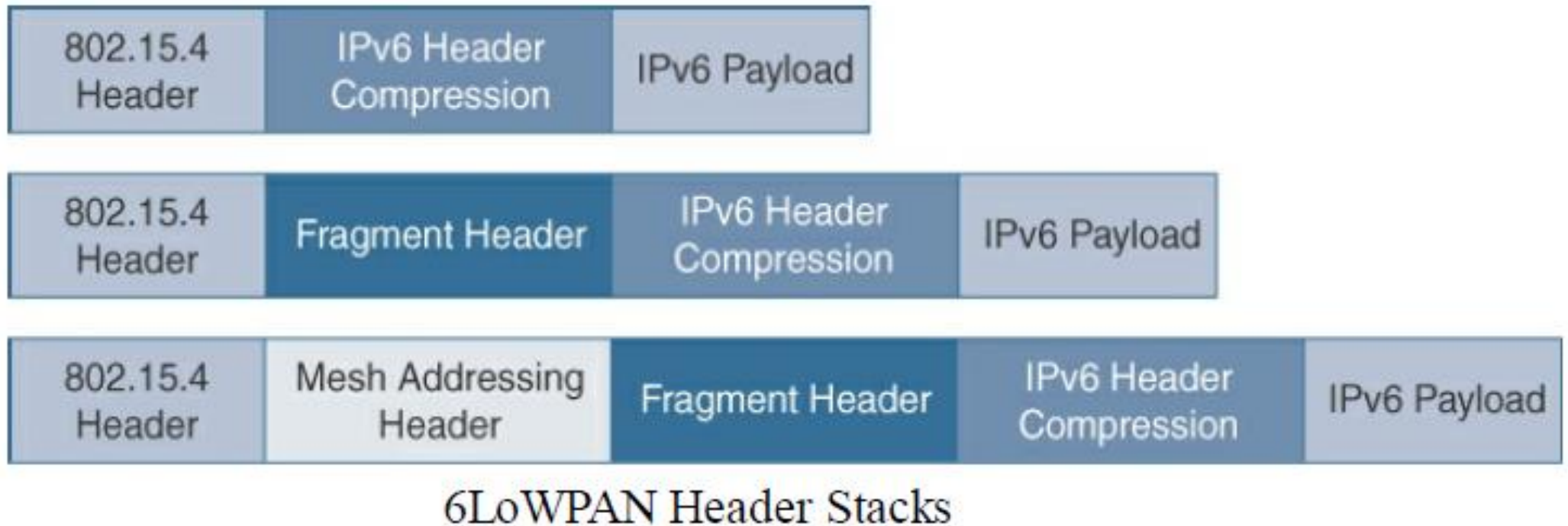
Figure shows the sub headers related to compression, fragmentation, and mesh addressing.

From 6LoWPAN to 6Lo:

Header Compression:

➢ **IPv6 header compression for 6LoWPAN was defined initially in RFC 4944 and subsequently updated by RFC 6282.**

➢ **This capability shrinks the size of IPv6's 40-byte headers and User Datagram Protocol's (UDP's) 8-byte headers down as low as 6 bytes combined in some cases.**

➢ **Note that header compression for 6LoWPAN is only defined for an IPv6 header and not IPv4.**

➢ **The 6LoWPAN protocol does not support IPv4, and, in fact, there is no standardized IPv4 adaptation layer for IEEE 802.15.4.**

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

Header Compression:

➢ **6LoWPAN header compression is stateless.**

➢ **number of factors affect the amount of compression, such as implementation of RFC 4944 versus RFC 6922, whether UDP is included, and various IPv6 addressing scenarios.**

➢ **At a high level, 6LoWPAN works by taking advantage of shared information known by all nodes from their participation in the local network.**

➢ **In addition, it omits some standard header fields by assuming commonly used values.**

➢ **Figure highlights an example that shows the amount of reduction that is possible with 6LoWPAN header compression.**

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

6LoWPAN Without Header Compression

127 Byte IEEE 802.15.4 Frame

| 1B | 40B | 8B | 53B |

802.15.4 Header | | IPv6 | UDP | Payload | FCS

↑
6LoWPAN Header

6LoWPAN With IPv6 and UDP Header Compression

127 Byte IEEE 802.15.4 Frame

| 2B | 4B | 108B |

802.15.4 Header | | UDP | Payload | FCS

↑
6LoWPAN Header
with Compressed
IPv6 Header

6LoWPAN Header Compression
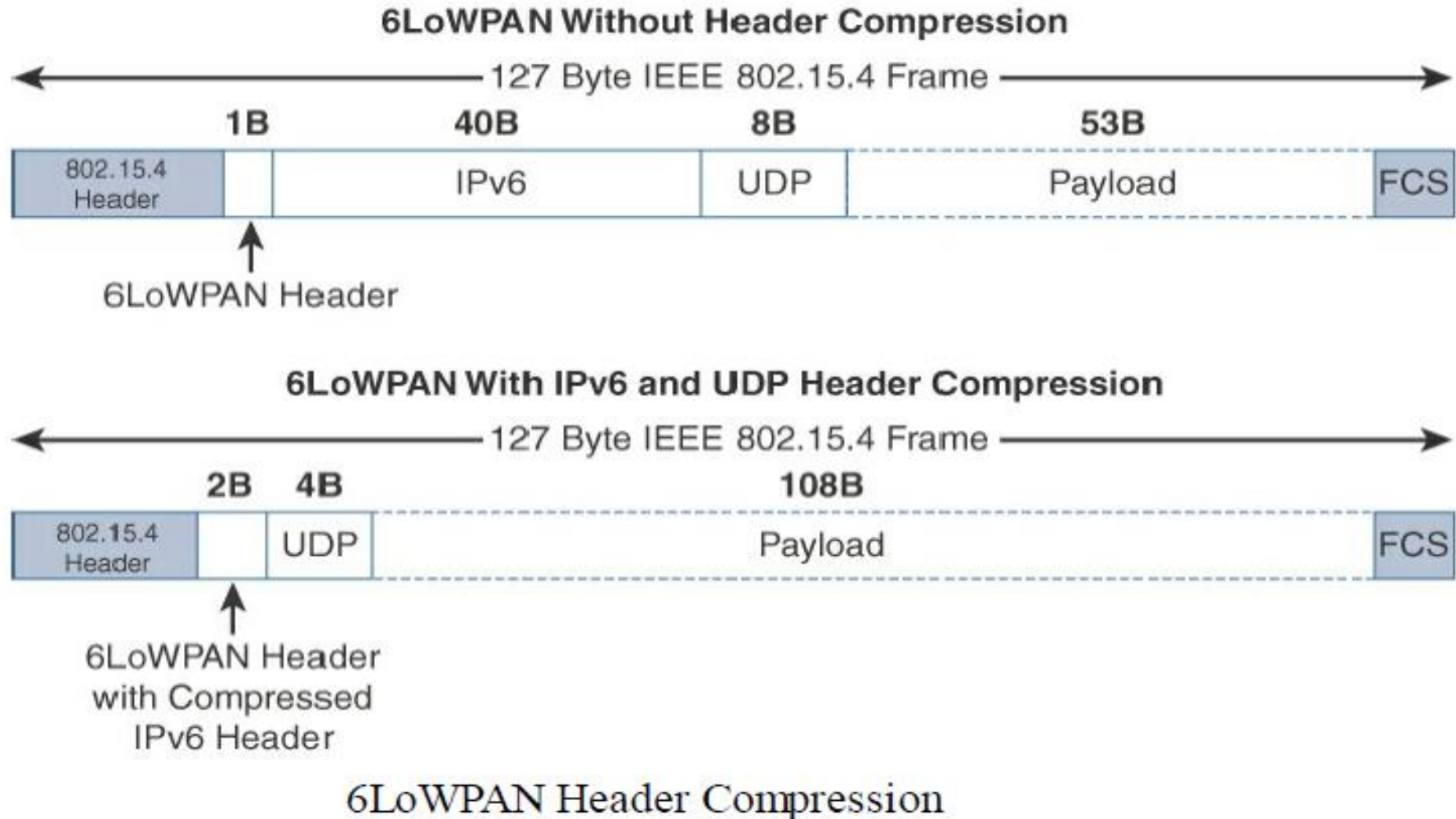
Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

Header Compression:

➢ **At the top , see a 6LoWPAN frame without any header compression enabled: The full 40-byte IPv6 header and 8-byte UDP header are visible.**

➢ **The 6LoWPAN header is only a single byte in this case.**

➢ **uncompressed IPv6 and UDP headers leave only 53 bytes of data payload out of the 127-byte maximum frame size in the case of IEEE 802.15.4.**

➢ **The bottom half of Figure shows a frame where header compression has been enabled for a best-case scenario.**

Module – 3 IP as the IoT Network Layer

From 6LoWPAN to 6Lo:

Header Compression:

➢ **The 6LoWPAN header increases to 2 bytes to accommodate the compressed IPv6 header, and UDP has been reduced in half, to 4 bytes from 8.**

➢ **Most importantly, the header compression has allowed the payload to more than double, from 53 bytes to 108 bytes, which is obviously much more efficient.**

➢ **Note that the 2-byte header compression applies to intra-cell communications, while communications external to the cell may require some field of the header to not be compressed.**

Module – 3 IP as the IoT Network Layer

Fragmentation:

➢   The **maximum transmission unit (MTU)** for an IPv6 network must be at least **1280 bytes**.

➢   The term MTU defines the **size of the largest protocol** data unit that can be passed.

➢   For IEEE 802.15.4, **127 bytes** is the MTU.

➢   In IPv6, with a much larger MTU, is carried inside the 802.15.4 frame with a **much smaller one**.

➢   To remedy this situation, **large IPv6 packets** must be fragmented across multiple 802.15.4 frames at **Layer 2**.

Module – 3 IP as the IoT Network Layer

Fragmentation:

➤ **The fragment header utilized by 6LoWPAN is composed of three primary fields:**

   • **Datagram Size, Datagram Tag, and Datagram Offset.**

➤ **The 1-byte Datagram Size field specifies the total size of the unfragmented payload.**

➤ **Datagram Tag identifies the set of fragments for a payload.**

➤ **Finally, the Datagram Offset field delineates how far into a payload a particular fragment occurs.**

➤ **Figure provides an overview of a 6LoWPAN fragmentation header.**

Module – 3 IP as the IoT Network Layer

Fragmentation:

6LoWPAN Fragmentation Header
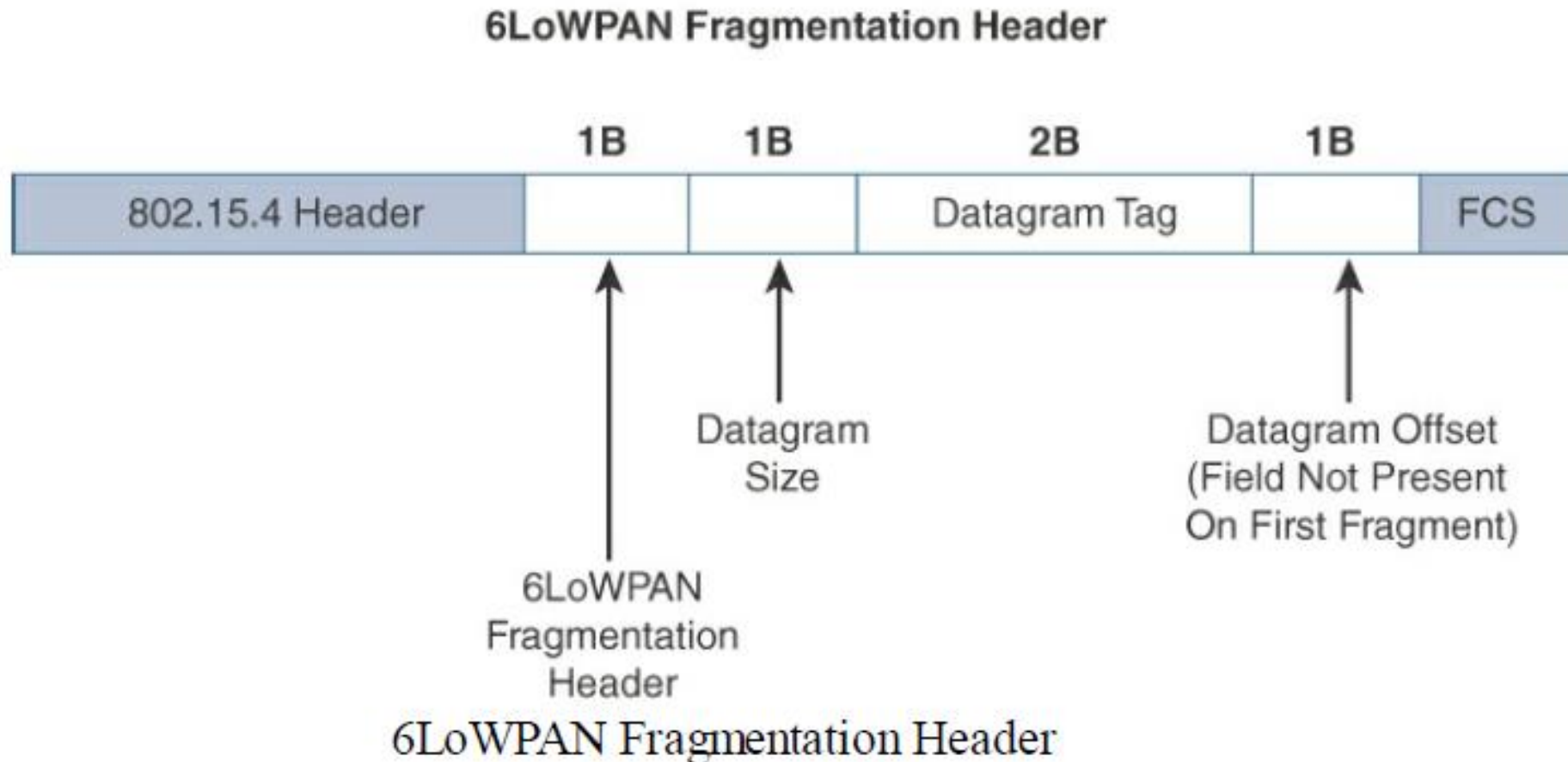
Module – 3 IP as the IoT Network Layer

Fragmentation:

➢ **In Figure, the 6LoWPAN fragmentation header field itself uses a unique bit value to identify that the subsequent fields behind it are fragment fields as opposed to another capability, such as header compression.**

➢ **Also, in the first fragment, the Datagram Offset field is not present because it would simply be set to 0.**

➢ **This results in the first fragmentation header for an IPv6 payload being only 4 bytes long.**

➢ **The remainder of the fragments have a 5-byte header field so that the appropriate offset can be specified.**

Mesh Addressing:

➤ The purpose of the 6LoWPAN mesh addressing **function** is to **forward packets** over multiple hops.

➤ Three fields are defined for this header: **Hop Limit, Source Address, and Destination Address.**

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

➤ Each hop **decrements** this **value by 1** as it is forwarded.

➤ Once the value hits **0, it is dropped** and no longer forwarded.

Mesh Addressing:

➢ **The Source Address and Destination Address fields for mesh addressing are IEEE 802.15.4 addresses indicating the endpoints of an IP hop details the 6LoWPAN mesh addressing header fields.**
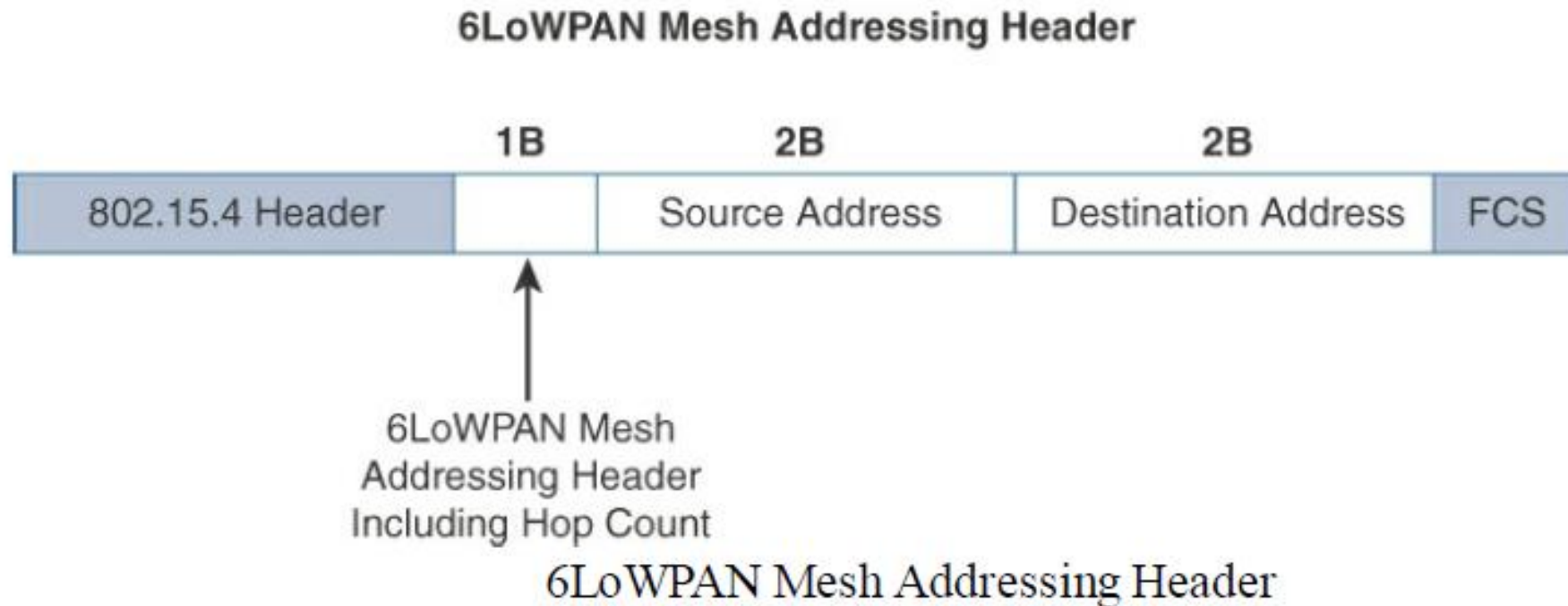
6LoWPAN Mesh Addressing Header

	1B	2B	2B	
802.15.4 Header		Source Address	Destination Address	FCS

↑
6LoWPAN Mesh
Addressing Header
Including Hop Count

6LoWPAN Mesh Addressing Header

Mesh Addressing:

➢ **Note that the mesh addressing header is used in a single IP subnet and is a Layer 2 type of routing known as mesh-under.**

➢ **RFC 4944 only provisions the function in this case as the definition of Layer 2 mesh routing specifications was outside the scope of the 6LoWPAN working group, and the IETF doesn't define "Layer 2 routing."**

➢ **An implementation performing Layer 3 IP routing does not need to implement a mesh addressing header unless required by a given technology profile.**

Mesh Addressing:

➢ **Note that the mesh addressing header is used in a single IP subnet and is a Layer 2 type of routing known as mesh-under.**

➢ **RFC 4944 only provisions the function in this case as the definition of Layer 2 mesh routing specifications was outside the scope of the 6LoWPAN working group, and the IETF doesn't define "Layer 2 routing."**

➢ **An implementation performing Layer 3 IP routing does not need to implement a mesh addressing header unless required by a given technology profile.**

Module – 3 IP as the IoT Network Layer

Mesh-Under Versus Mesh-Over Routing:

➢ **For network technologies such as IEEE 802.15.4, IEEE 802.15.4g, and IEEE 1901.2a that support mesh topologies and operate at the physical and data link layers, two main options exist for establishing reachability and forwarding packets.**

➢ **first option, mesh-under, the routing of packets is handled at the 6LoWPAN adaptation layer.**

➢ **The other option, known as "mesh-over" or "route-over," utilizes IP routing for getting packets to their destination**

Mesh-Under Versus Mesh-Over Routing:

➢ The term **mesh-under** is used because **multiple link layer hops** can be used to complete a **single IP hop**.

➢ Nodes have a **Layer 2 forwarding** table that they consult **to route** the packets to their **final destination** within the mesh.

➢ An **edge gateway** terminates the **mesh-under domain**.

➢ The edge gateway must also **implement** a mechanism to **translate** between the **configured Layer 2 protocol** and any IP routing mechanism implemented on other **Layer 3 IP** interfaces.

Mesh-Under Versus Mesh-Over Routing:

➢ **In mesh-over or route-over scenarios, IP Layer 33 routing is utilized for computing reachability and then getting packets forwarded to their destination, either inside or outside the mesh domain.**

➢ **Each full-functioning node acts as an IP router, so each link layer hop is an IP hop.**

➢ **When a LoWPAN has been implemented using different link layer technologies, a mesh-over routing setup is useful.**

➢ **While traditional IP routing protocols can be used, a specialized routing protocol for smart objects, such as RPL(Routing Protocol for Low Power and Lossy Networks), is recommended**

Dr. Syed Mustafa, HKBKCE.   60

Module – 3 IP as the IoT Network Layer

6Lo Working Group:

➢ **The charter of the 6Lo working group, now called the IPv6 over Networks of Resource-Constrained Nodes, is to facilitate the IPv6 connectivity over constrained-node networks.**

➢ **In particular, this working group is focused on the following:**

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

Module – 3 IP as the IoT Network Layer

6Lo Working Group:

➢ **For example, this includes:**

1. **IPv6 over Bluetooth Low Energy**

2. **Transmission of IPv6 packets over near-field communication**

3. **IPv6 over 802.11ah**

4. **Transmission of IPv6 packets over DECT Ultra Low Energy**

5. **Transmission of IPv6 packets on WIA-PA (Wireless Networks for Industrial Automation–Process Automation)**

6. **Transmission of IPv6 over Master Slave/Token Passing (MS/TP)**

Module – 3 IP as the IoT Network Layer

6Lo Working Group:

➢ **Information and data models such as MIB modules:**

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

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

- **For example, this includes RFC 7400, "6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)."**

➢ **Informational and maintenance publications needed for the IETF specifications in this area**

6Lo Working Group:

In summary,

➢ **the 6Lo working group is standardizing the 6LoWPAN adaptation layer that initially focused on the IEEE 802.15.4 Layer 2 protocol to others that are commonly found with constrained nodes.**

➢ **In fact, based on the work of the 6LoWPAN working group and now the 6Lo working group, the 6LoWPAN adaptation layer is becoming the de factor standard for connecting constrained nodes in IoT networks.**

6TiSCH:

➤ **IEEE 802.15.4e, Time-Slotted Channel Hopping (TSCH), is an add-on to the Media Access Control (MAC) portion of the IEEE 802.15.4 standard, with direct inheritance from other standards, such as WirelessHART and ISA100.11a.**

➤ **Devices implementing IEEE 802.15.4e TSCH communicate by following a Time Division Multiple Access (TDMA) schedule.**

➤ **An allocation of a unit of bandwidth or time slot is scheduled between neighbor nodes.**

➤ **This allows the programming of predictable transmissions and enables deterministic, industrial-type applications.**

Module – 3 IP as the IoT Network Layer

6TiSCH:

➢ **To standardize IPv6 over the TSCH mode of IEEE 802.15.4e (known as 6TiSCH), the IETF formed the 6TiSCH working group.**

➢ **This working group works on the architecture, information model, and minimal 6TiSCH configuration, leveraging and enhancing work done by the 6LoWPAN working group, RoLL (Routing over Low-Power and Lossy Networks) working group, and CoRE(Constrained Restful Environments) working group.**

➢ **The RoLL working group focuses on Layer 3 routing for constrained networks..**

6TiSCH:

➤ **An important element specified by the 6TiSCH working group is 6top, a sublayer that glues together the MAC layer and 6LoWPAN adaptation layer.**

➤ **This sublayer provides commands to the upper network layers, such as RPL.**

➤ **In return, these commands enable functionalities including network layer routing decisions, configuration, and control procedures for 6TiSCH schedule management.**

➤ **The IEEE 802.15.4e standard defines a time slot structure, but it does not mandate a scheduling algorithm for how the time slots are utilized.**

➤ **Scheduling is critical because it can affect throughput, latency, and power consumption**

6TiSCH:

➢ **Figure shows where6top resides in relation to IEEE 802.15.4e, 6LoWPAN HC, and IPv6.**
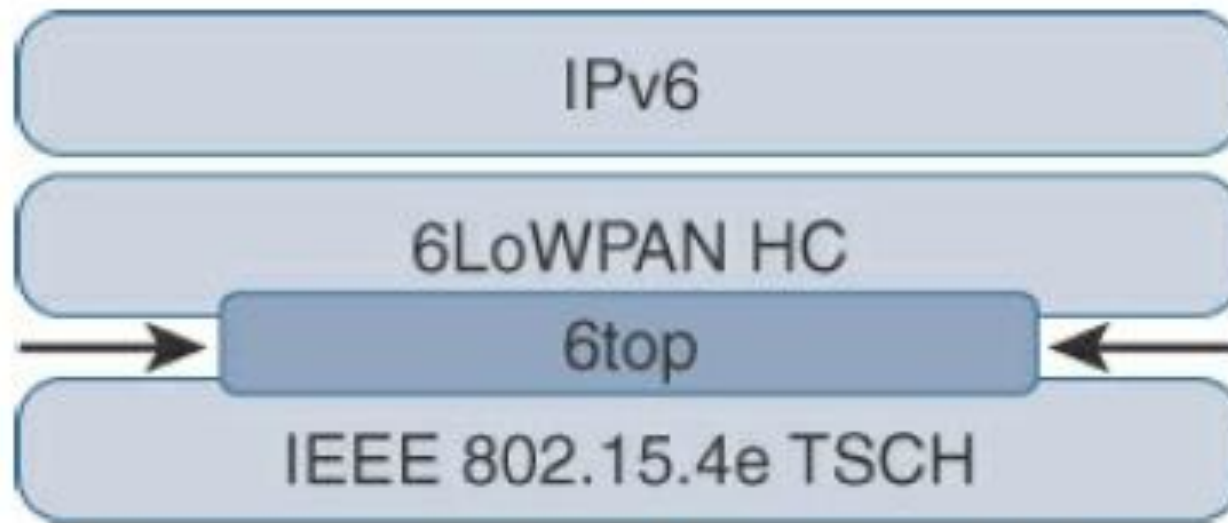
Location of 6TiSCH's 6top Sublayer

6TiSCH:

➤ **Schedules in 6TiSCH are broken down into cells.**

➤ **A cell is simply a single element in the TSCH schedule that can be allocated for unidirectional or bidirectional communications between specific nodes.**

➤ **Nodes only transmit when the schedule dictates that their cell is open for communication.**

6TiSCH:

The 6TiSCH architecture defines four schedule management mechanisms:

1. Static scheduling:

➢ **All nodes in the constrained network share a fixed schedule.**

➢ **Cells are shared, and nodes contend for slot access in a slotted aloha manner.**

➢ **Slotted aloha is a basic protocol for sending data using time slot boundaries when communicating over a shared medium.**

➢ **Static scheduling is a simple scheduling mechanism that can be used upon initial implementation or as a fallback in the case of network malfunction.**

6TiSCH:

The 6TiSCH architecture defines four schedule management mechanisms:

2. Neighbor-to-neighbor scheduling:

➢ **A schedule is established that correlates with the observed number of transmissions between nodes.**

➢ **Cells in this schedule can be added or deleted as traffic requirements and bandwidth needs change.**

6TiSCH:

The 6TiSCH architecture defines four schedule management mechanisms:

3. Remote monitoring and scheduling management:

➢ **Time slots and other resource allocation are handled by a management entity that can be multiple hops away.**

➢ **The scheduling mechanism leverages 6top and even CoAP(Constrained Application Protocol) in some scenarios.**

➢ **This scheduling mechanism provides quite a bit of flexibility and control in allocating cells for communication between nodes.**

6TiSCH:

The 6TiSCH architecture defines four schedule management mechanisms:

4. Hop-by-hop scheduling:

➢ **A node reserves a path to a destination node multiple hops away by requesting the allocation of cells in a schedule at each intermediate node hop in the path.**

➢ **The protocol that is used by a node to trigger this scheduling mechanism is not defined at this point.**

Module – 3 IP as the IoT Network Layer

6TiSCH:

➤ **In addition to schedule management functions, the 6TiSCH architecture also defines three different forwarding models.**

➤ **Forwarding is the operation performed on each packet by a node that allows it to be delivered to a next hop or an upper-layer protocol.**

➤ **The forwarding decision is based on a preexisting state that was learned from a routing computation.**

6TiSCH:

There are three 6TiSCH forwarding models:

1. **Track Forwarding (TF):**

- **This is the simplest and fastest forwarding model.**

- **A "track" in this model is a unidirectional path between a source and a destination.**

- **This track is constructed by pairing bundles of receive cells in a schedule with a bundle of receive cells set to transmit.**

- **So, a frame received within a particular cell or cell bundle is switched to another cell or cell bundle.**

- **This forwarding occurs regardless of the network layer protocol.**

6TiSCH:

There are three 6TiSCH forwarding models:

2. Fragment forwarding (FF):

- **This model takes advantage of 6LoWPAN fragmentation to build a Layer 2 forwarding table.**

- **IPv6 packets can get fragmented at the 6LoWPAN sublayer to handle the differences between IEEE 802.15.4 payload size and IPv6 MTU.**

- **Additional headers for RPL source route information can further contribute to the need for fragmentation.**

Module – 3 IP as the IoT Network Layer

6TiSCH:

There are three 6TiSCH forwarding models:

2. Fragment forwarding (FF):

- **with FF, a mechanism is defined where the first fragment is routed based on the IPv6 header present.**

- **The 6LoWPAN sublayer learns the next-hop selection of this first fragment, which is then applied to all subsequent fragments of that packet.**

- **Otherwise, IPv6 packets undergo hop-by-hop reassembly.**

- **This increases latency and can be power- and CPU-intensive for a constrained node.**

6TiSCH:

There are three 6TiSCH forwarding models:

3. IPv6 Forwarding (6F):

- **This model forwards traffic based on its IPv6 routing table.**

- **Flows of packets should be prioritized by traditional QoS (quality of service) and RED (random early detection) operations.**

- **QoS is a classification scheme for flows based on their priority, and RED is a common congestion avoidance mechanism.**

RPL:

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

➢ **A new routing protocol should be developed for use by IP smart objects, given the characteristics and requirements of constrained networks.**

➢ **This new distance-vector routing protocol was named the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL).**

➢ **The RPL specification was published as RFC 6550 by the RoLL working group.**

Module – 3 IP as the IoT Network Layer

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **In an RPL network, each node acts as a router and becomes part of a mesh network.**

➢ **Routing is performed at the IP layer.**

➢ **Each node examines every received IPv6 packet and determines the next-hop destination based on the information contained in the IPv6 header.**

➢ **No information from the MAC-layer header is needed to perform next-hop determination.**

Dr. Syed Mustafa, HKBKCE. 80

Module – 3 IP as the IoT Network Layer

RPL (Routing Protocol for Low Power and Lossy Networks):

To cope with the **constraints of computing** and **memory** that are common characteristics of constrained nodes, the protocol defines **two modes**:

1. **Storing mode**:

    - All nodes contain the **full routing table** of the RPL domain.

    - Every node knows how to **directly reach** every other node.

Module – 3 IP as the IoT Network Layer

RPL (Routing Protocol for Low Power and Lossy Networks):

2. Non-storing mode:

- Only the **border router(s)** of the RPL domain contain(s) the **full routing table.**

- All other nodes in the domain **only maintain their list of parents** and use this as a list of default routes **toward the border router.**

- This abbreviated routing table **saves memory space and CPU.**

- When communicating in non-storing mode, a node always **forwards** its packets to the **border router**, which knows how to ultimately reach the final destination.

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **RPL is based on the concept of a directed acyclic graph (DAG).**

➢ **A DAG is a directed graph where no cycles exist.**

➢ **This means that from any vertex or point in the graph, cannot follow an edge or a line back to this same point.**

➢ **All of the edges are arranged in paths oriented toward and terminating at one or more root nodes.**

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **A basic RPL process involves building a destination-oriented directed acyclic graph (DODAG).**

➢ **A DODAG is a DAG rooted to one destination.**

➢ **In RPL, this destination occurs at a border router known as the DODAG root.**

➢ **DAG has multiple roots, whereas the DODAG has just one.**

➢ **In a DODAG, each node maintains up to three parents that provide a path to the root.**

➢ **Typically, one of these parents is the preferred parent, that is the preferred next hop for upward routes toward the root.**

RPL (Routing Protocol for Low Power and Lossy Networks):
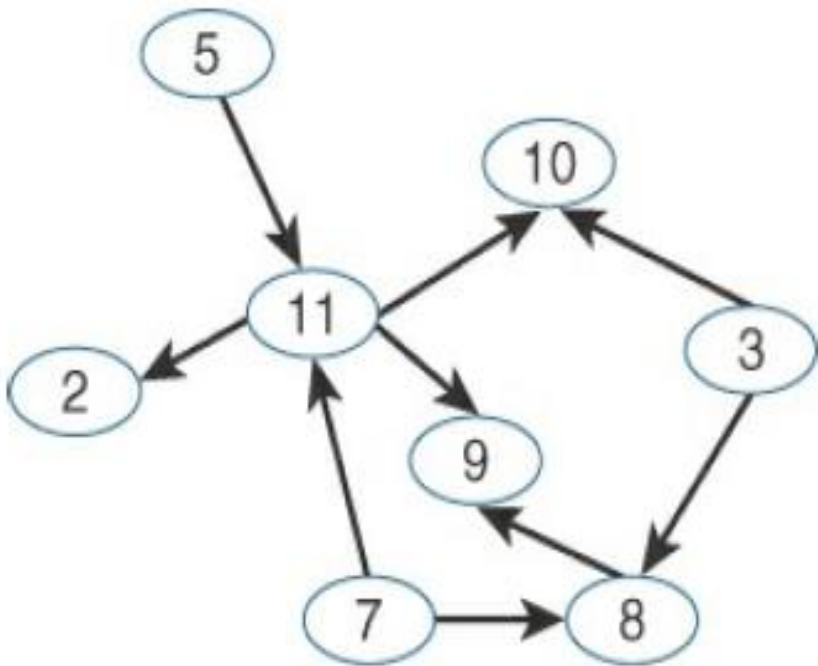
Example of a Directed Acyclic Graph (DAG)
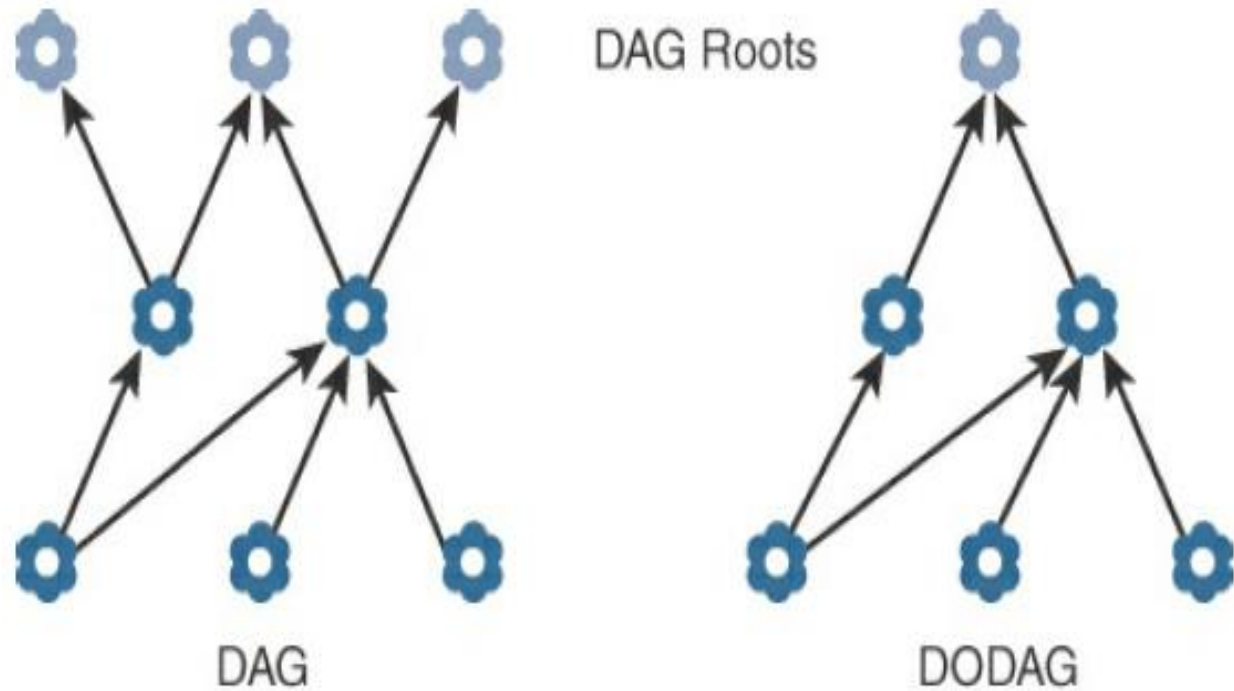
DAG and DODAG Comparison

DAG

DODAG

DAG Roots

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **The routing graph created by the set of DODAG parents across all nodes defines the full set of upward routes.**

➢ **Upward routes in RPL are discovered and configured using DAG Information Object (DIO) messages.**

➢ **Nodes listen to DIOs to handle changes in the topology that can affect routing.**

➢ **The information in DIO messages determines parents and the best path to the DODAG root.**

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **Nodes establish downward routes by advertising their parent set toward the DODAG root using a Destination Advertisement Object (DAO) message.**

➢ **DAO messages allow nodes to inform their parents of their presence and reachability to descendants.**

➢ **In the case of the non-storing mode of RPL, nodes sending DAO messages report their parent sets directly to the DODAG root (border router), and only the root stores the routing information.**

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **The root uses the information to then determine source routes needed for delivering IPv6 datagrams to individual nodes downstream in the mesh.**

➢ **For storing mode, each node keeps track of the routing information that is advertised in the DAO messages.**

➢ **While this is more power- and CPU intensive for each node, the benefit is that packets can take shorter paths between destinations in the mesh.**
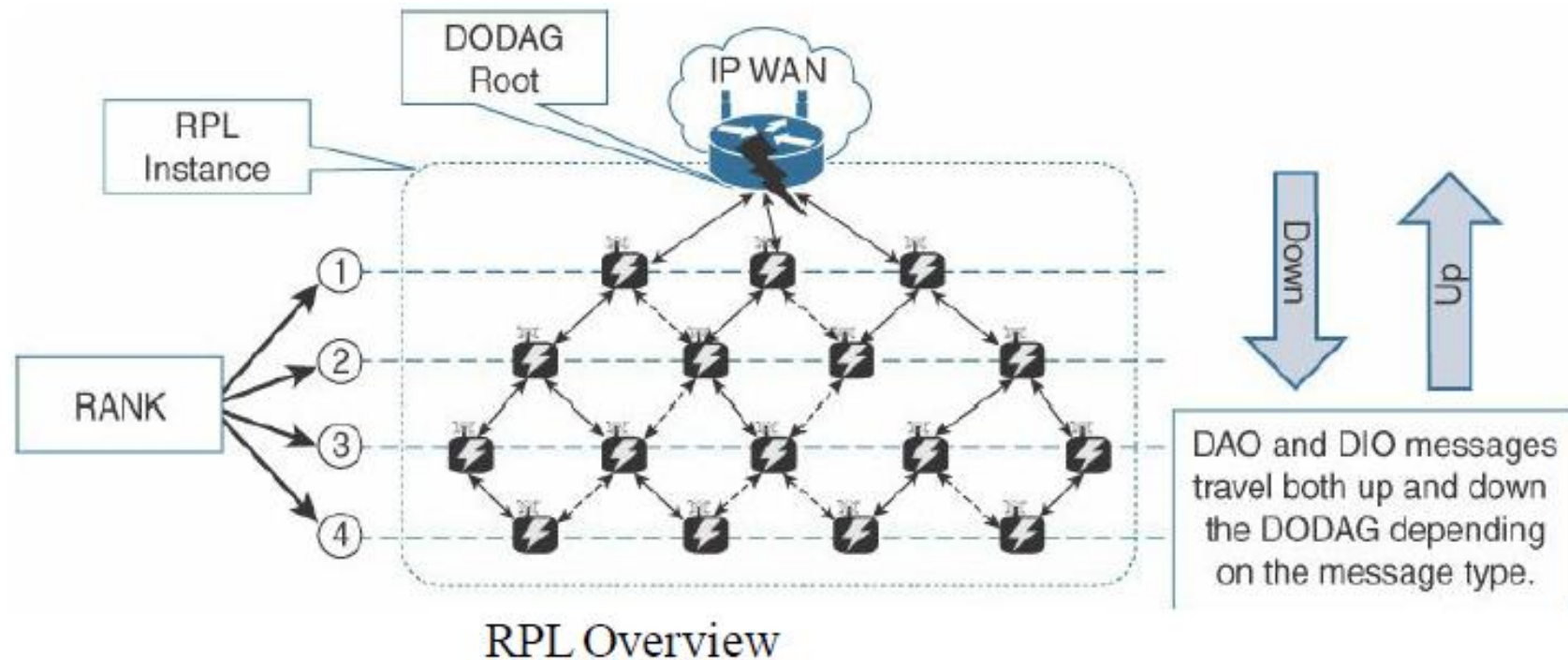
RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **The nodes can make their own routing decisions.**

➢ **In non-storing mode, on the other hand, all packets must go up to the root to get a route for moving downstream.**

➢ **RPL messages, such as DIO and DAO, run on top of IPv6.**

➢ **These messages exchange and advertise downstream and upstream routing information between a border router and the nodes under it.**

Module – 3 IP as the IoT Network Layer

RPL (Routing Protocol for Low Power and Lossy Networks):

➢ **Figure shows how DAO and DIO messages move both up and down the DODAG, depending on the exact message type.**

RPL Overview

RPL (Routing Protocol for Low Power and Lossy Networks):

Objective Function (OF):

➢ **An objective function (OF) defines how metrics are used to select routes and establish a node's rank.**

➢ **Standards such as RFC 6552 and 6719 have been published to document OFs specific to certain use cases and node types.**

➢ **For example, nodes implementing an OF based on RFC 6719's Minimum Expected Number of Transmissions (METX) advertise the METX among their parents in DIO messages. Whenever a node establishes its rank, it simply sets the rank to the current minimum METX among its parents.**

RPL (Routing Protocol for Low Power and Lossy Networks):

Rank:

➢ **The rank is a rough approximation of how "close" a node is to the root and helps avoid routing loops and the count-to-infinity problem.**

➢ **Nodes can only increase their rank when receiving a DIO message with a larger version number.**

➢ **Nodes may decrease their rank whenever they have established lower-cost routes.**

➢ **While the rank and routing metrics are closely related, the rank differs from routing metrics in that it is used as a constraint to prevent routing loops.**

RPL (Routing Protocol for Low Power and Lossy Networks):

RPL Headers:

➤ **Specific network layer headers are defined for datagrams being forwarded within an RPL domain.**

➤ **One of the headers is standardized in RFC 6553, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams," and the other is discussed in RFC 6554, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)."**

➤ **RFC 6553 defines a new IPv6 option, known as the RPL option.**

Module – 3 IP as the IoT Network Layer

RPL (Routing Protocol for Low Power and Lossy Networks):

RPL Headers:

➢ **The RPL option is carried in the IPv6 Hop-by-Hop header.**

➢ **The purpose of this header is to leverage data-plane packets for loop detection in a RPL instance.**

➢ **DODAGs only have single paths and should be loop free.**

➢ **RFC 6554 specifies the Source Routing Header (SRH) for use between RPL routers.**

➢ **A border router or DODAG root inserts the SRH when specifying a source route to deliver datagrams to nodes downstream in the mesh network.**

RPL (Routing Protocol for Low Power and Lossy Networks):

Metrics:

➢ **RPL defines a large and flexible set of new metrics and constraints for routing in RFC 6551.**

➢ **Developed to support powered and battery-powered nodes, RPL offers a far more complete set than any other routing protocol.**

➢ **Some of the RPL routing metrics and constraints defined in RFC 6551 include the following:**

RPL (Routing Protocol for Low Power and Lossy Networks):

Metrics:

➢ **Expected Transmission Count (ETX):** Assigns a discrete value to the number of transmissions a node expects to make to deliver a packet.

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

➢ **Latency:** Varies depending on power conservation. Paths with a lower latency are preferred.

RPL (Routing Protocol for Low Power and Lossy Networks):

Metrics:

➢ **Link Quality Level: Measures the reliability of a link by taking into account packet error rates caused by factors such as signal attenuation and interference.**

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

RPL (Routing Protocol for Low Power and Lossy Networks):

Metrics:

➤ **Node State and Attribute:** **Identifies nodes that function as traffic aggregators and nodes that are being impacted by high workloads. High workloads could be indicative of nodes that have incurred high CPU or low memory states.**

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

➤ **Throughput:** **Provides the amount of throughput for a node link. Often, nodes conserving power use lower throughput.**

Module – 3 IP as the IoT Network Layer

Authentication and Encryption on Constrained Nodes:

IETF working groups that are focused on the security: ACE and DICE.

ACE:

Much like the RoLL working group, the Authentication and Authorization for Constrained Environments (ACE) working group is tasked with evaluating the applicability of existing authentication and authorization protocols and documenting their suitability for certain constrained-environment use cases.

ACE:

➢ **Once the candidate solutions are validated, the ACE working group will focus its work on CoAP with the Datagram Transport Layer Security (DTLS) protocol.**

➢ **The ACE working group expects to produce a standardized solution for authentication and authorization that enables authorized access (Get, Put, Post, Delete) to resources identified by a URI and hosted on a resource server in constrained environments.**

Module – 3 IP as the IoT Network Layer

ACE:

➤ **An unconstrained authorization server performs mediation of the access.**

➤ **Aligned with the initial focus, access to resources at a resource server by a client device occurs using CoAP and is protected by DTLS(Datagram Transport Layer Security).**

Module – 3 IP as the IoT Network Layer

DICE:

➢ **New generations of constrained nodes implementing an IP stack over constrained access networks are expected to run an** **optimized IP protocol stack****.**

➢ **For example, when implementing UDP at the transport layer, the IETF Constrained Application Protocol (CoAP) should be** **used at the application layer****.**

➢ **In constrained environments secured by DTLS,** **CoAP** **can be used to** **control resources** **on a device.**

DICE:

➢ **The DTLS in Constrained Environments (DICE) working group focuses on implementing the DTLS transport layer security protocol in these environments.**

➢ **The first task of the DICE working group is to define an optimized DTLS profile for constrained nodes.**

➢ **In addition, the DICE working group is considering the applicability of the DTLS record layer to secure multicast messages and investigating how the DTLS handshake in constrained environments can get optimized.**

Module – 3 IP as the IoT Network Layer

Profiles and Compliances:

Internet Protocol for Smart Objects (IPSO) Alliance:

➤ **Established in 2008, the Internet Protocol for Smart Objects (IPSO) initially focused on promoting IP as the premier solution for smart objects communications.**

➤ **Today, it is more focused on how to use IP, with the IPSO Alliance organizing interoperability tests between alliance members to validate that IP for smart objects can work together and properly implement industry standards.**

Module – 3 IP as the IoT Network Layer

Profiles and Compliances:

Internet Protocol for Smart Objects (IPSO) Alliance:

➢ **The IPSO Alliance does not define technologies,, but it documents the use of IP-based technologies for various IoT use cases and participates in educating the industry.**

➢ **As the IPSO Alliance declares in its value and mission statement, it wants to ensure that "engineers and product builders will have access to the necessary tools for 'how to build the IoT RIGHT'.**

Wi-SUN Alliance:

➢ **The Wi-SUN Alliance is an example of efforts from the industry to define a communication profile that applies to specific physical and data link layer protocols.**

➢ **Currently, Wi-SUN's main focus is on the IEEE 802.15.4g protocol and its support for multiservice and secure IPv6 communications with applications running over the UDP transport layer.**

➢ **The utilities industry is the main area of focus for the Wi-SUN Alliance.**

Module – 3 IP as the IoT Network Layer

Wi-SUN Alliance:

➢ **The Wi-SUN field area network (FAN) profile enables smart utility networks to provide resilient, secure, and cost-effective connectivity with extremely good coverage in a range of topographic environments, from dense urban neighborhoods to rural areas.**

Module – 3 IP as the IoT Network Layer

Thread:

- **A group of companies involved with smart object solutions for consumers created the Thread Group.**

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

- **The wireless technology used by Thread is IEEE 802.15.4, which is different from Wi-SUN's IEEE 802.15.4g**

Module – 3 IP as the IoT Network Layer

IPv6 Ready Logo:

➢ **Initially, the IPv6 Forum ensured the promotion of IPv6 around the world.**

➢ **Once IPv6 implementations became widely available, the need for interoperability and certification led to the creation of the IPv6 Ready Logo program.**

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

Module – 3 IP as the IoT Network Layer

IPv6 Ready Logo:

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

➢ **These certifications have industry-wide recognition, and many products are already certified.**

➢ **An IPv6 certification effort specific to IoT is currently under definition for the program.**

Module – 3 Application Protocols for IoT

This chapter focuses on how higher-layer IoT protocols are transported.

Specifically, this chapter includes the following sections:

The Transport Layer:

> IP-based networks use either TCP or UDP.

> However, the constrained nature of IoT networks requires a closer look at the use of these traditional transport mechanisms.

IoT Application Transport Methods:

> This section explores the various types of IoT application data and the ways this data can be carried across a network.

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.**

- ➢ **we 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.**

Module – 3 Application Protocols for IoT

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.**

Module – 3 Application Protocols for IoT

➢ **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.**

Module – 3 Application Protocols for IoT

➢ **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.**

Module – 3 Application Protocols for IoT

➢ **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), may not cope well with supporting large numbers of TCP sessions.**

Module – 3 Application Protocols for IoT

➢ **new IoT application protocol, such as Constrained Application Protocol (CoAP), almost always uses UDP and why implementations of industrial 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) application layer protocol, a popular protocol for reading smart meters in the utilities space, is the de facto standard in Europe.**

Module – 3 Application Protocols for IoT

For example, if we compare the transport of DLMS/COSEM over a cellular network versus an LLN deployment, we should consider the following:

➢ **Select TCP** for **cellular networks** because these networks are typically **more robu**st 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.**

For example, if we compare the transport of DLMS/COSEM over a cellular network versus an LLN deployment, we should consider the following:

➢ **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 separate 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.**

Module – 3 Application Protocols for IoT

For example, if we compare the transport of DLMS/COSEM over a cellular network versus an LLN deployment, we should consider the following:

➢ When transferring **large amounts** of DLMS/COSEM data, **cellular links** are preferred to optimize each open association.

➢ **Smaller amounts of data** can be handled efficiently 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.

Module – 3 Application Protocols for IoT

Summary:

➤ **TCP and UDP are the two main choices at the transport layer for the TCP/IP protocol.**

➤ **The performance and scalability of IoT constrained devices and networks is impacted by which one of these is selected.**

Module – 3 Application Protocols for IoT

IoT Application Transport Methods:

The following categories of IoT application protocols and their transport methods are explored in the following sections:

1. **Application layer protocol not present**: **In this case, the data payload is directly transported on top of the lower layers. No application layer protocol is used.**

2. **Supervisory control and data acquisition (SCADA):** **SCADA is one of the most common industrial protocols in the world, but it was developed long before the days of IP, and it has been adapted for IP networks.**

Module – 3 Application Protocols for IoT

IoT Application Transport Methods:

3. Generic web-based protocols:

- Generic protocols, such as **Ethernet, Wi-Fi, and 4G/LTE**, are found on many consumer- and enterprise-class IoT devices that communicate over non-constrained networks.

4. IoT application layer protocols:

- IoT application layer protocols are devised to **run on constrained nodes** with a small compute footprint and are well adapted to the **network bandwidth constraints** on cellular or satellite links or constrained 6LoWPAN networks.

- **Message Queuing Telemetry Transport (MQTT) and** Constrained Application Protocol (CoAP), are two well-known examples of IoT application layer protocols

Module – 3 Application Protocols for IoT

1. **Application Layer Protocol Not Present:**

➢ **IETF RFC 7228 devices defined as class 0** send or receive only a few bytes of data.

➢ For myriad reasons, such as **processing capability, power constraints, and cost**, these devices do not implement a **fully structured network protocol stack**, such as **IP, TCP, or UDP**, or even an application layer protocol.

➢ Class 0 devices are usually **simple smart objects** that are severely constrained.

➢ Implementing a **robust protocol** stack is usually not useful and sometimes not even possible with the limited available resources.

Module – 3 Application Protocols for IoT

1. **Application Layer Protocol Not Present:**

➢ **For example, consider low-cost temperature and relative humidity (RH)sensors sending data over an LPWA LoRaWAN infrastructure.**

➢ **Temperature is represented as 2 bytes and RH as another 2 bytes of data.**

➢ **This small data payload is directly transported on top of the LoRaWAN MAC layer, without the use of TCP/IP.**

Module – 3 Application Protocols for IoT

1. **Application Layer Protocol Not Present:**

➢ **Different kinds of temperature sensors will report temperature data in varying formats.**

➢ **A temperature value will always be present in the data transmitted by each sensor, but decoding this data will be vendor specific.**

➢ **If we scale this scenario out across hundreds or thousands of sensors, the problem of allowing various applications to receive and interpret temperature values delivered in different formats becomes increasingly complex.**

Module – 3 Application Protocols for IoT

1.   **Application Layer Protocol Not Present:**

➤ **The solution to this problem is to use an IoT data broker, as detailed in Figure**

➤ **An IoT data broker is a piece of middleware that standardizes sensor output into a common format that can then be retrieved by authorized applications**
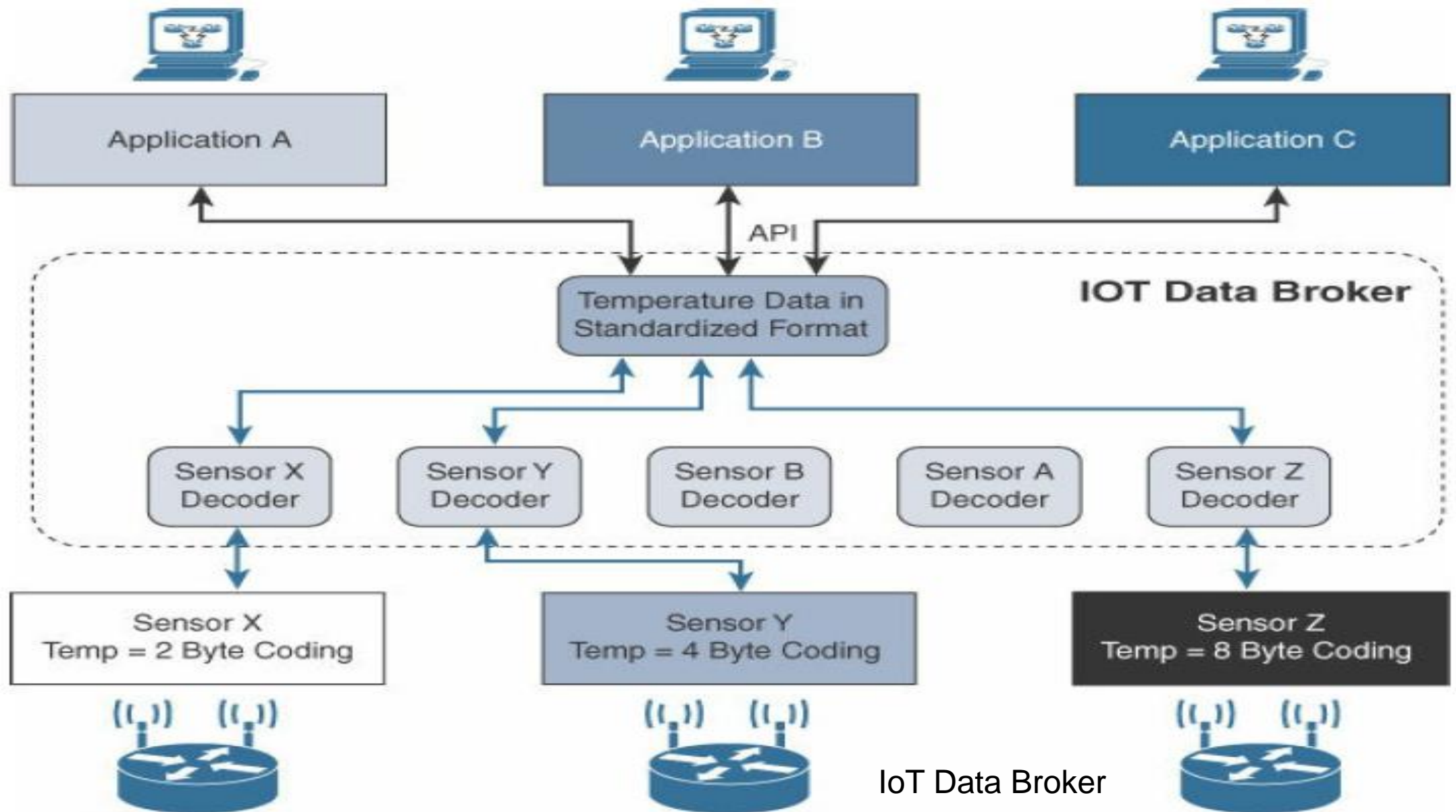
IoT Data Broker

Module – 3 Application Protocols for IoT

1. **Application Layer Protocol Not Present:**

➢ **In Figure, Sensors X, Y, and Z are all temperature sensors, but their output is encoded differently.**

➢ **The IoT data broker understands the different formats in which the temperature is encoded and is therefore able to decode this data into a common, standardized format.**

➢ **Applications A, B, and C in Figure can access this temperature data without having to deal with decoding multiple temperature data formats.**

2. **SCADA**-supervisory control and data acquisition:

➢ **Designed decades ago, SCADA is an automation control system that was initially implemented without IP over serial links, before being adapted to Ethernet and IPv4.**

➢ **At a high level, SCADA systems collect sensor data and telemetry from remote devices, while also providing the ability to control them.**

➢ **Used in today's networks, SCADA systems allow global, real-time, data-driven decisions to be made about how to improve business processes.**

Module – 3 Application Protocols for IoT

2. **SCADA**-supervisory control and data acquisition:

➢ **SCADA networks can be found across various industries, but mainly concentrated in the utilities and manufacturing/industrial verticals.**

➢ **Within these specific industries, SCADA commonly uses certain protocols for communications between devices and applications.**

➢ **For example, Modbus and its variants are industrial protocols used to monitor and program remote devices via a master/slave relationship.**

Module – 3 Application Protocols for IoT

2. **SCADA**-supervisory control and data acquisition:

➢ **Modbus is also found in building management, transportation, and energy applications.**

➢ **The DNP3 and International Electro technical Commission (IEC) 60870-5-101 protocols are found mainly in the utilities industry, along with DLMS/COSEM and ANSI C12 for advanced meter reading (AMR).**

2. SCADA-supervisory control and data acquisition:

Adapting SCADA for IP:

- **In the 1990s, the rapid adoption of Ethernet networks in the industrial world drove the evolution of SCADA application layer protocols.**

- **The support of legacy industrial protocols over IP networks, protocol specifications were updated and published, documenting the use of IP for each protocol.**

2. SCADA -supervisory control and data acquisition:

Adapting SCADA for IP:

This included assigning TCP/UDP port numbers to the protocols, such as the following:

1. **DNP3 (adopted by IEEE 1815-2012) specifies the use of TCP or UDP on port 20000 for transporting DNP3 messages over IP.**

2. **The Modbus messaging service utilizes TCP port 502.**

3. **IEC 60870-5-104 is the evolution of IEC 60870-5-101 serial for running over Ethernet and IPv4 using port 2404.**

2. SCADA-**supervisory control and data acquisition**:

4. DLMS **User Association specified a communication profile based on TCP/IP in the DLMS/COSEM Green Book (Edition 5 or higher), or in the IEC 62056-53 and IEC 62056-47 standards, allowing data exchange via IP and port 4059.**

- **These legacy serial protocols have adapted and evolved to utilize IP and TCP/UDP as both networking and transport mechanisms.**

- **This has allowed utilities and other companies to continue leveraging their investment in equipment and infrastructure, supporting these legacy protocols with modern IP networks.**

2. **SCADA**-**supervisory control and data acquisition**:

➢ **SCADA protocols, DNP3 is based on a master/slave relationship.**

➢ **master refers to typically a powerful computer located in the control center of a utility, and a slave is a remote device with computing resources found in a location such as a substation.**

➢ **DNP3 refers to slaves specifically as outstations.**

➢ **Outstations monitor and collect data from devices that indicate their state, such as whether a circuit breaker is on or off, and take measurements, including voltage, current, temperature, and so on..**

2. **SCADA**-**supervisory control and data acquisition**:

➢ **The IEEE 1815-2012 specification describes how the DNP3 protocol implementation must be adapted to run either over TCP (recommended) or UDP.**

➢ **This specification defines connection management between the DNP3 protocol and the IP layers, as shown in Figure.**

➢ **Connection management links the DNP3 layers with the IP layers in addition to the configuration parameters and methods necessary for implementing the network connection.**

2. **SCADA**-**supervisory control and data acquisition**:

➢ **The IP layers appear transparent to the DNP3 layers as each piece of the protocol stack in one station logically communicates with the respective part in the other.**

➢ **This means that the DNP3 endpoints or devices are not aware of the underlying IP transport that is occurring**
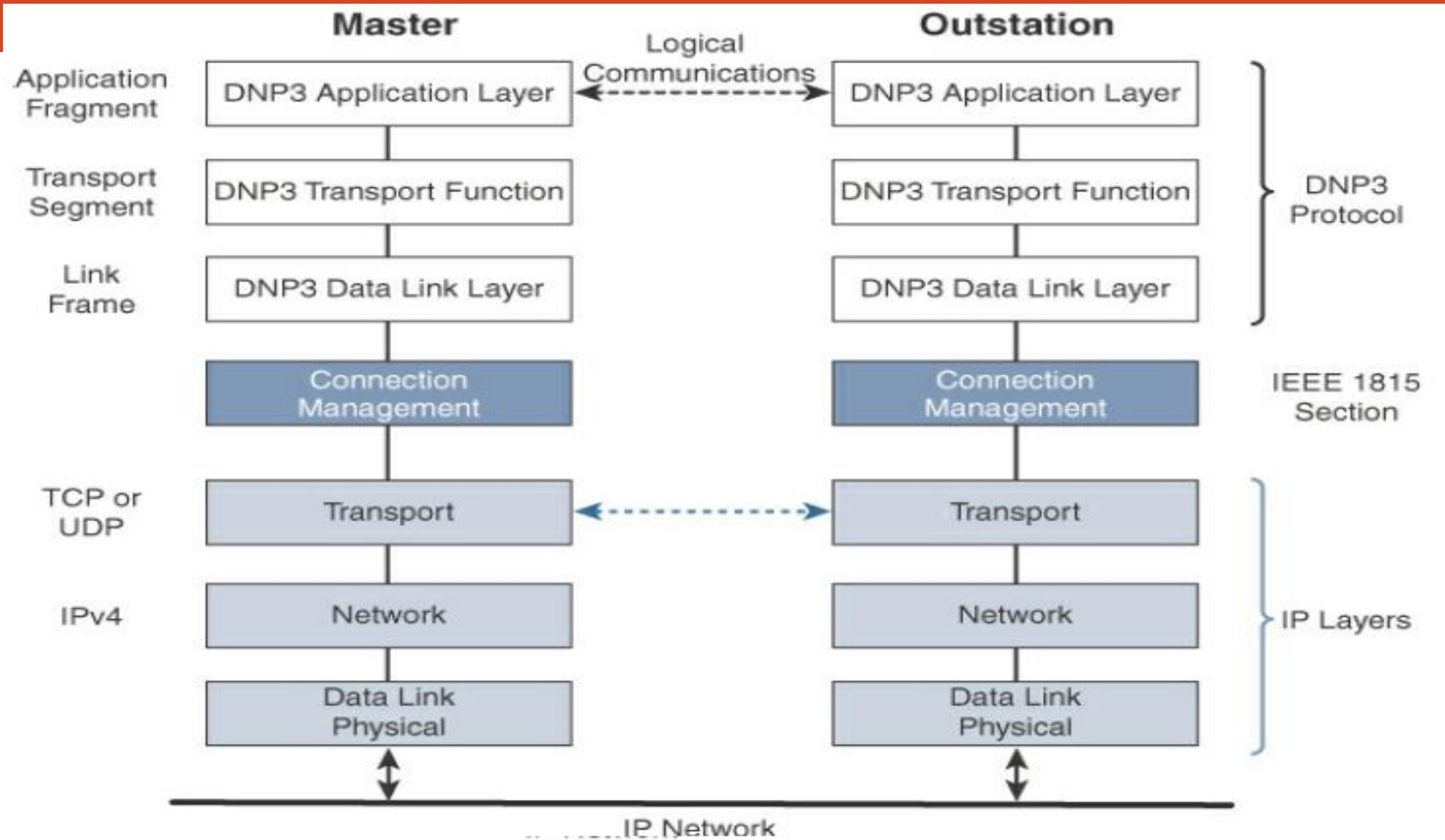
Protocol Stack for Transporting Serial DNP3 SCADA over IP

2. **SCADA**-**supervisory control and data acquisition**:

➢ **In Figure, the master side initiates connections by performing a TCP active open.**

➢ **The outstation listens for a connection request by performing a TCP passive open.**

➢ **Dual endpoint is defined as a process that can both listen for connection requests and perform an active open on the channel if required.**

2. **SCADA-supervisory control and data acquisition**:

➢ **Master stations may parse multiple DNP3 data link layer frames from a single** UDP datagram, while DNP3 data link layer frames **cannot span multiple** UDP datagrams.

➢ **Single or multiple** connections to the master may get established while a TCP keepalive **timer monitors** the status of the connection.

➢ **Keepalive** messages are implemented as DNP3 data link layer **status requests**.

➢ If a response is not received to a **keepalive message**, the connection is **deemed broken**, and the appropriate action is taken.

Tunneling Legacy SCADA over IP Networks:

➤ **Deployments of legacy industrial protocols, such as DNP3 and other SCADA protocols, in modern IP networks call for flexibility when integrating several generations of devices or operations that are tied to various releases and versions of application servers.**

➤ **Transport of the original serial protocol over IP can be achieved either by tunneling using raw sockets over TCP or UDP or by installing an intermediate device that performs protocol translation between the serial protocol version and its IP implementation**

Tunneling Legacy SCADA over IP Networks:

➤ **A raw socket connection simply denotes that the serial data is being packaged directly into a TCP or UDP transport.**

➤ **A socket in this instance is a standard application programming interface (API) composed of an IP address and a TCP or UDP port that is used to access network devices over an IP network.**

➤ **Figure details raw socket scenarios for a legacy SCADA server trying to communicate with remote serial devices.**

Tunneling Legacy SCADA over IP Networks:

Raw Socket TCP or UDP Scenarios for Legacy Industrial Serial Protocols
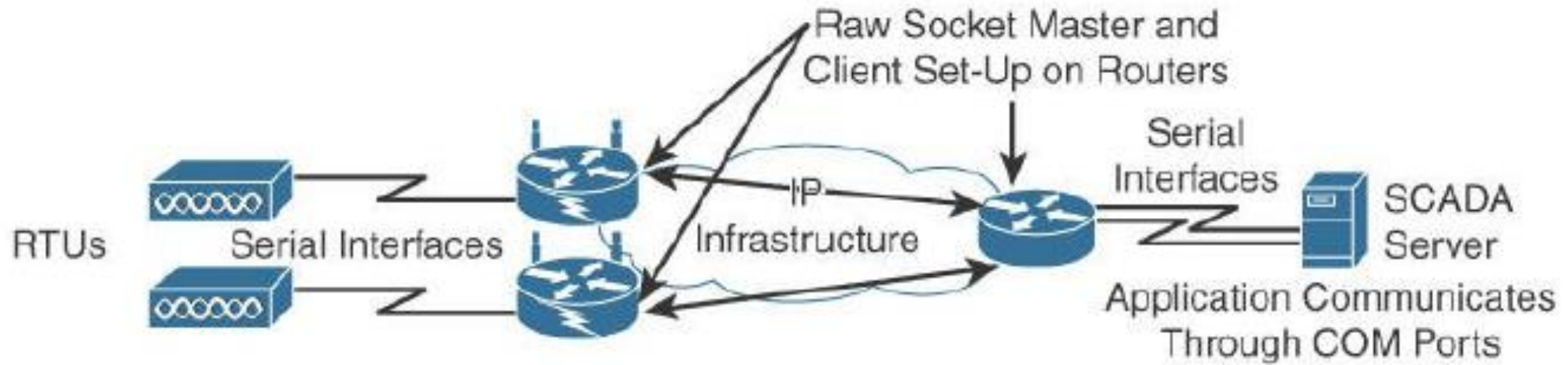
Scenario A: Raw Socket between Routers – no change on SCADA server

Tunneling Legacy SCADA over IP Networks:

Raw Socket TCP or UDP Scenarios for Legacy Industrial Serial Protocols

Tunneling Legacy SCADA over IP Networks:
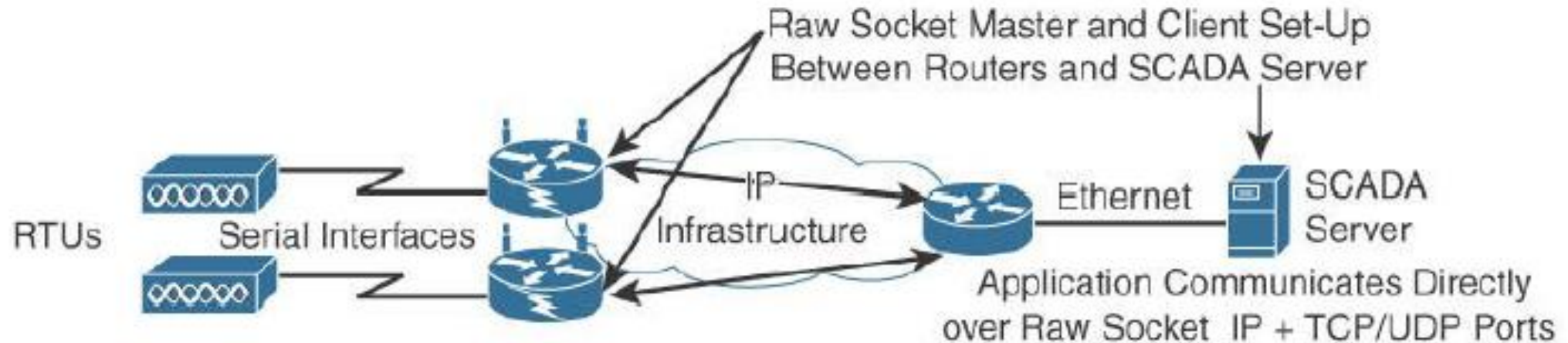
Raw Socket TCP or UDP Scenarios for Legacy Industrial Serial Protocols

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

Tunneling Legacy SCADA over IP Networks:

➢ **In all the scenarios in Figure, that routers connect via serial interfaces to the remote terminal units (RTUs), which are often associated with SCADA networks.**

➢ **An RTU is a multipurpose device used to monitor and control various systems, applications, and devices managing automation.**

➢ **From the master/slave perspective, the RTUs are the slaves. Opposite the RTUs in each Figure scenario is a SCADA server, or master, that varies its connection type.**

Tunneling Legacy SCADA over IP Networks:

➢ **In Scenario A in Figure, both the SCADA server and the RTUs have a direct serial connection to their respective routers.**

➢ **The routers terminate the serial connections at both ends of the link and use raw socket encapsulation to transport the serial payload over the IP network.**

Tunneling Legacy SCADA over IP Networks:

➢ **Scenario B has a small change on the SCADA server side.**

➢ **A piece of software is installed on the SCADA server that maps the serial COM ports to IP ports.**

➢ **This software is commonly referred to as an IP/serial redirector.**

➢ **The IP/serial redirector in essence terminates the serial connection of the SCADA server and converts it to a TCP/IP port using a raw socket connection.**

Module – 3 Application Protocols for IoT

Tunneling Legacy SCADA over IP Networks:

➢ **In Scenario C in Figure, the SCADA server supports native raw socket capability.**

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

Module – 3 Application Protocols for IoT

SCADA Protocol Translation:

➢ **An alternative to a raw socket connection for transporting legacy serial data across an IP network is protocol translation.**

➢ **With protocol translation, the legacy serial protocol is translated to a corresponding IP version.**

➢ **For example, Figure shows two serially connected DNP3 RTUs and two master applications supporting DNP3 over IP that control and pull data from the RTUs.**

➢ **The IoT gateway in this figure performs a protocol translation function that enables communication between the RTUs and servers, despite the fact that a serial connection is present on one side and an IP connection is used on the other.**

➢ **By running protocol translation, the IoT gateway connected to the RTUs in Figure is implementing a computing function close to the edge of the network.**

➢ **Adding computing functions close to the edge helps scale distributed intelligence in IoT networks**

Module – 3 Application Protocols for IoT

SCADA Protocol Translation:

➢ **This can be accomplished by offering computing resources on IoT gateways or routers, as shown in this protocol translation example.**

➢ **Alternatively, this can also be performed directly on a node connecting multiple sensors.**

➢ **In either case, this is referred to as fog computing.**

SCADA Protocol Translation:

DNP3 Protocol Translation
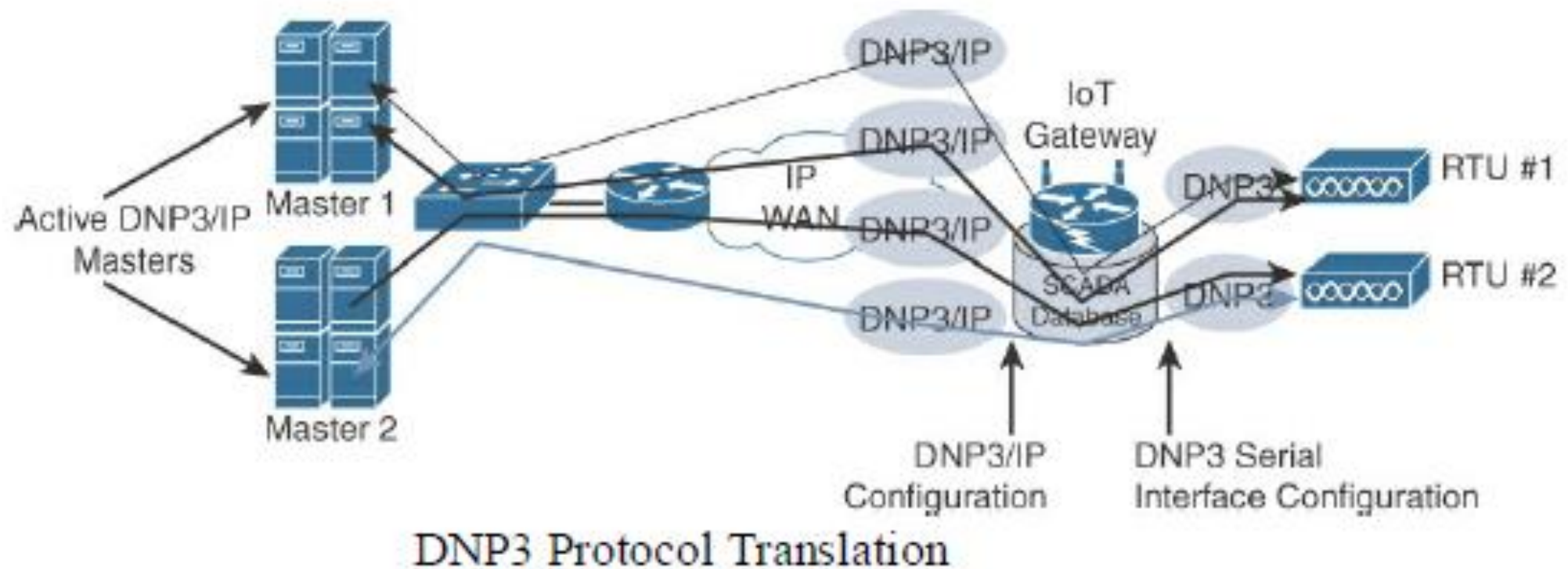
Module – 3 Application Protocols for IoT

SCADA Transport over LLNs with MAP-T:

➢ **Due to the constrained nature of LLNs, the implementation of industrial protocols should at a minimum be done over UDP.**

➢ **This in turn requires that both the application servers and devices support and implement UDP.**

➢ **While the long-term evolution of SCADA and other legacy industrial protocols is to natively support IPv6, but most of the industrial devices supporting IP today support IPv4 only.**

Module – 3 Application Protocols for IoT

SCADA Transport over LLNs with MAP-T:

➢ **When deployed over LLN subnetworks that are IPv6 only, a transition mechanism, such as MAP-T (Mapping of Address and Port using Translation, RFC 7599), needs to be implemented.**

➢ **This allows the deployment to take advantage of native IPv6 transport transparently to the application and devices.**

SCADA Transport over LLNs with MAP-T:

➢ **Figure depicts a scenario in which a legacy endpoint is connected across an LLN running 6LoWPAN to an IP-capable SCADA server.**

➢ **The legacy endpoint could be running various industrial and SCADA protocols, including DNP3/IP, Modbus/TCP, or IEC 60870-5-104.**

➢ **In this scenario, the legacy devices and the SCADA server support only IPv4 (typical in the industry today).**

➢ **However, IPv6 (with 6LoWPAN and RPL) is being used for connectivity to the endpoint**

SCADA Transport over LLNs with MAP-T:

➢ **6LoWPAN is a standardized protocol designed for constrained networks, but it only supports IPv6.**

➢ **In this situation, the end devices, the endpoints, and the SCADA server support only IPv4, but the network in the middle supports only IPv6.**

➢ **The solution to this problem is to use the protocol known as MAP-T makes the appropriate mappings between IPv4 and the IPv6 protocols.**

➢ **This allows legacy IPv4 traffic to be forwarded across IPv6 networks.**

SCADA Transport over LLNs with MAP-T:
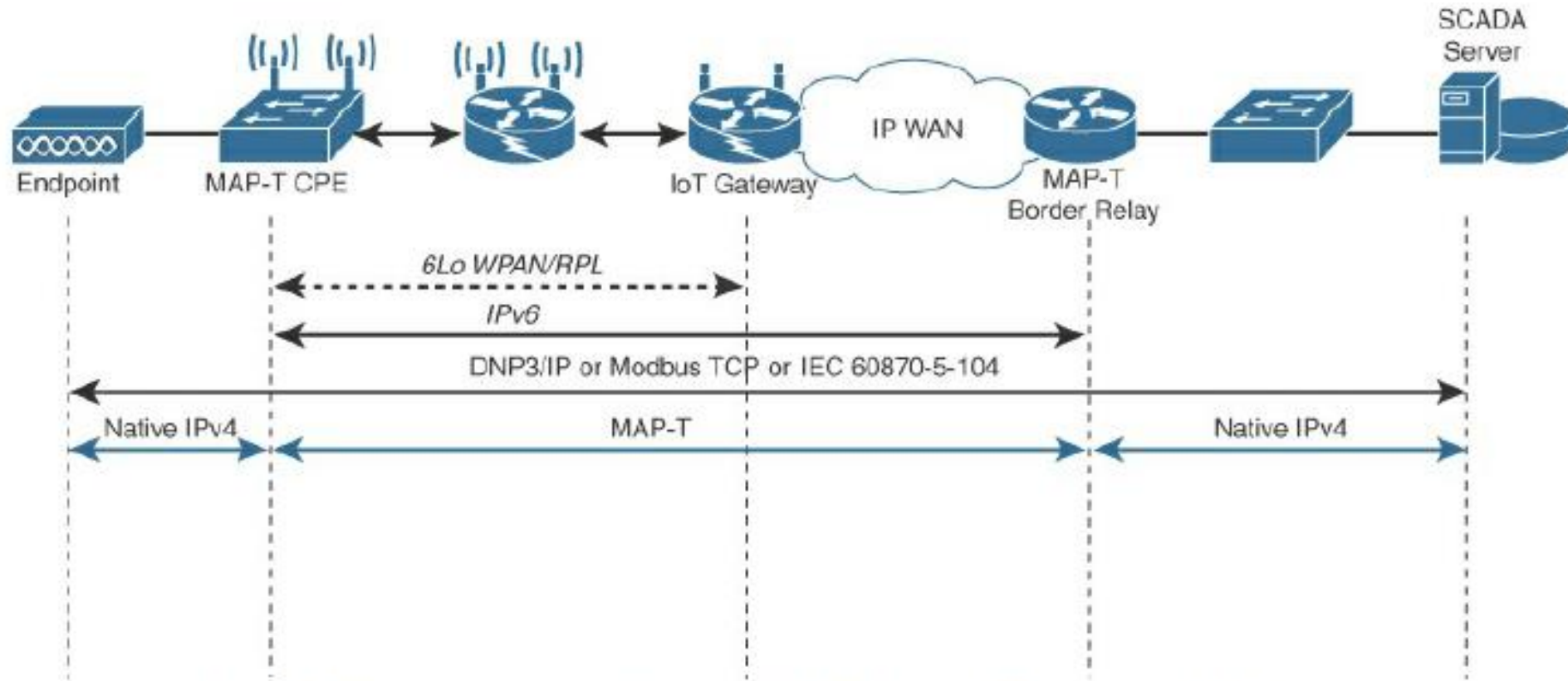
DNP3 Protocol over 6LoWPAN Networks with MAP-T

Dr. Syed Mustafa, HKBKCE.

SCADA Transport over LLNs with MAP-T:

➤ In Figure, the IPv4 endpoint on the left side is connected to a **Customer Premise Equipment (CPE)** device.

➤ The **MAP-T CPE device** has an IPv6 connection to the **RPL mesh**.

➤ On the right side, a **SCADA server with native IPv4** support connects to a **MAP-T border gateway**.

➤ The MAP-T CPE device and MAP-T border gateway are responsible for the **MAP-T conversion from IPv4 to IPv6.**

Module – 3 Application Protocols for IoT

Generic Web-Based Protocols:

➢ **On non-constrained networks, such as Ethernet, Wi-Fi, or 3G/4G cellular, where bandwidth is not perceived as a potential issue, data payloads based on a verbose data model representation, including XML or JavaScript Object Notation (JSON), can be transported over HTTP/HTTPS or WebSocket.**

➢ **This allows implementers to develop their IoT applications in contexts similar to web applications.**

Module – 3 Application Protocols for IoT

Generic Web-Based Protocols:

➤ The **HTTP/HTTPS client/server** model serves as the foundation for the World Wide Web.

➤ Recent evolutions of **embedded web server software** with advanced features are now implemented with **very little memory** (in the range of tens of kilobytes in some cases).

➤ This enables the **use of embedded web services** software on some constrained devices.

Module – 3 Application Protocols for IoT

Generic Web-Based Protocols:

➢ **Interactions between real-time communication tools powering collaborative applications, such as voice and video, instant messaging, chat rooms, and IoT devices, are also emerging.**

➢ **This is driving the need for simpler communication systems between people and IoT devices.**

➢ **One protocol that addresses this need is Extensible Messaging and Presence Protocol (XMPP).**

Module – 3 Application Protocols for IoT

IoT Application Layer Protocols:

➢ **IoT industry is working on new lightweight protocols that are better suited to large numbers of constrained nodes and networks.**

➢ **Two of the most popular protocols are CoAP and MQTT.**

➢ **Figure highlights their position in a common IoT protocol stack.**

➢ **In Figure, CoAP and MQTT are naturally at the top of this sample IoT stack, based on an IEEE 802.15.4 mesh network.**

Module – 3 Application Protocols for IoT

IoT Application Layer Protocols:

CoAP	MQTT
UDP	TCP
IPv6	
6LoWPAN	
802.15.4 MAC	
802.15.4 PHY	

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

Module – 3 Application Protocols for IoT

Constrained Application Protocol (CoAP):

➤ **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.

➤ The CoAP framework defines **simple and flexible** ways to manipulate **sensors and actuators** for data or device management.

Constrained Application Protocol (CoAP):

➤ **The IETF CoRE working group has published multiple standards-track specifications for CoAP, including the following:**

1. **RFC 6690**: Constrained RESTful Environments (CoRE) **Link Format**

2. **RFC 7252**: The Constrained Application Protocol (CoAP)

3. **RFC 7641:Observing** Resources in the Constrained Application Protocol (CoAP)

4. **RFC 7959:Block-Wise** Transfers in the Constrained Application Protocol (CoAP)

5. **RFC 8075**: Guidelines for **Mapping Implementations**: HTTP to the Constrained Application Protocol (CoAP)

Dr. Syed Mustafa, HKBKCE.
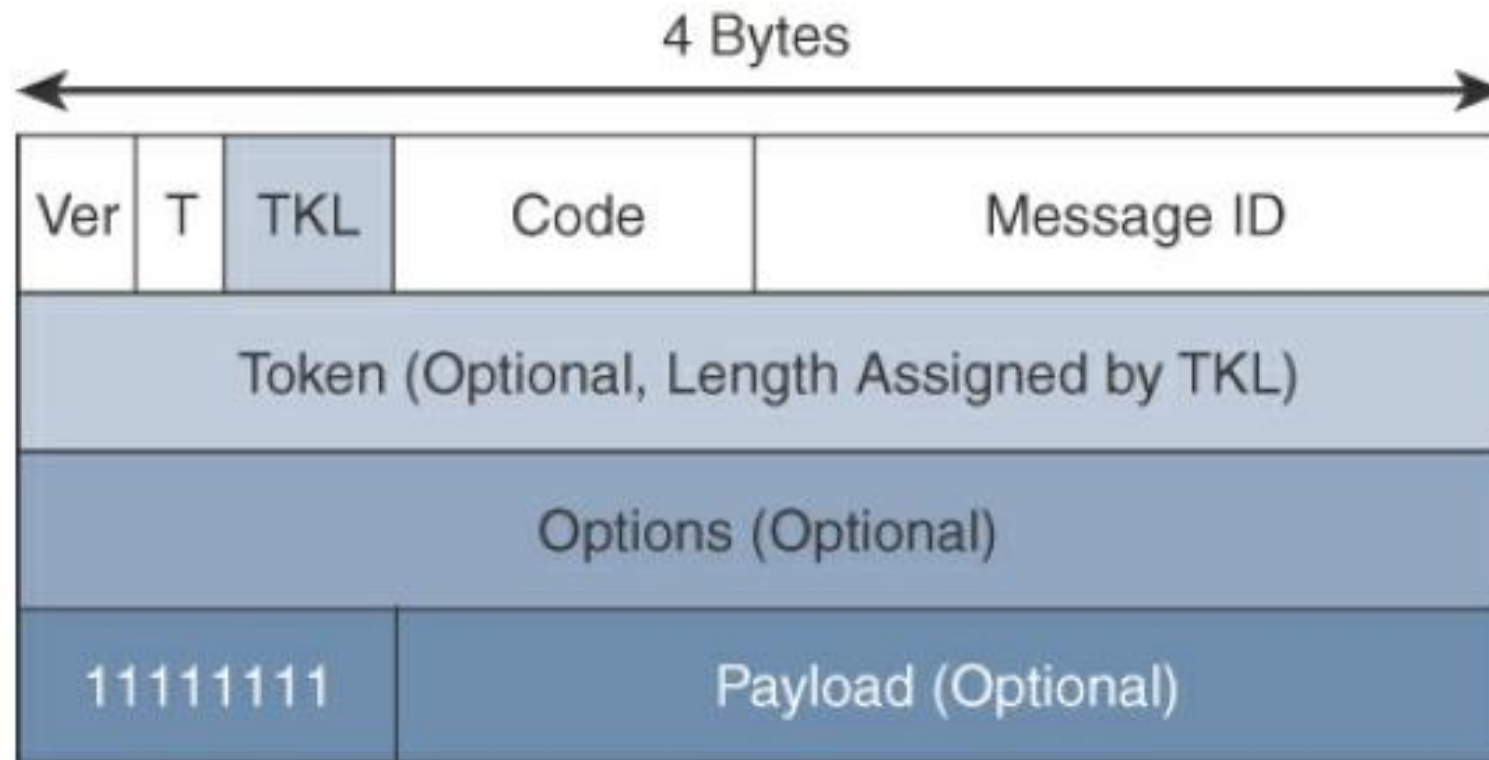
Module – 3 Application Protocols for IoT

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**).

➢ **CoAP over Short Message Service (SMS) as defined in Open Mobile Alliance for Lightweight Machine-to-Machine (LWM2M) for IoT device management is also being considered.**

➢ **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.**

Constrained Application Protocol (CoAP):

CoAP Message Format

Module – 3 Application Protocols for IoT

Constrained Application Protocol (CoAP):

➢ **CoAP message format is relatively simple and flexible.**

➢ **It allows CoAP to deliver low overhead, which is critical for constrained networks, while also being easy to parse and process for constrained devices.**

➢ **Table provides an overview of the various fields of a CoAP message.**

Constrained Application Protocol (CoAP):

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.

CoAP Message Fields

Module – 3 Application Protocols for IoT

Constrained Application Protocol (CoAP):

CoAP Message Field	Description
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.

CoAP Message Fields

Constrained Application Protocol (CoAP):

➢ **CoAP can run over IPv4 or IPv6.**

➢ **However, the message fit within a single IP packet and UDP payload to avoid fragmentation.**

➢ **For IPv6, with the default MTU size being 1280 bytes and allowing for no fragmentation across nodes, the maximum CoAP message size could be up to 1152 bytes, including 1024 bytes for the payload.**

➢ **In the case of IPv4, as IP fragmentation implementations should limit to more conservative values and set the IPv4 Don't Fragment (DF) bit.**
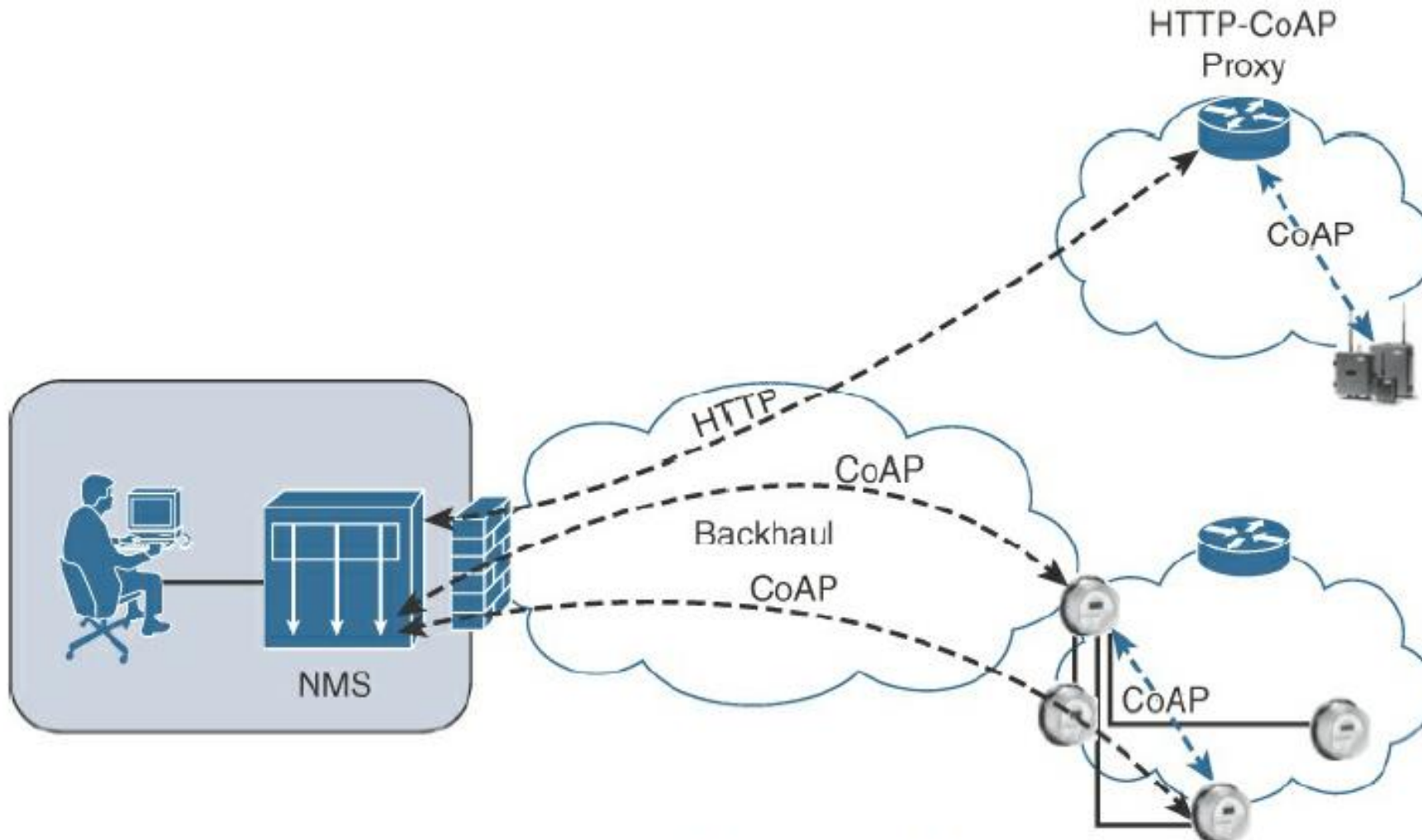
Module – 3 Application Protocols for IoT

Constrained Application Protocol (CoAP):

➤ **CoAP communications across an IoT infrastructure can take various paths.**

➤ **Connections can be between devices located on the same or different constrained networks or between devices and generic Internet or cloud servers, all operating over IP.**

➤ **Proxy mechanisms are also defined, and RFC 7252 details a basic HTTP mapping for CoAP.**

➤ **As both HTTP and CoAP are IP-based protocols, the proxy function can be located practically anywhere in the network, not necessarily at the border between constrained and non-constrained networks.**

Constrained Application Protocol (CoAP):

CoAP Communications in IoT Infrastructures

175

Constrained Application Protocol (CoAP):

➢ **Just like HTTP, CoAP is based on the REST architecture**, but with a "thing" acting as both the client and the server.

➢ **Through the exchange of asynchronous messages**, a client requests an action via a **method code** on a server resource.

➢ A **uniform resource identifier** (URI) localized on the server **identifies** this resource.

➢ The **server responds** with a response code that may include a **resource** representation.

➢ The CoAP **request/response** semantics include the methods **GET, POST, PUT, and DELETE.**

Module – 3 Application Protocols for IoT

Constrained Application Protocol (CoAP):

➢ **CoAP URI format is similar to HTTP/HTTPS.**

➢ **The coap/coaps URI scheme identifies a resource, including host information and optional UDP port, as indicated by the host and port parameters in the URI.**

```
coap-URI = "coap:" "//" host [":" port] path-abempty ["?" query]
coaps-URI = "coaps:" "//" host [":" port] path-abempty  ["?" query]
```

Constrained Application Protocol (CoAP):

- **CoAP defines four types of messages**:

    confirmable, non-confirmable, acknowledgement, and reset.

- **Method codes and response codes included in some of these messages make them carry requests or responses.**

- **CoAP code, method and response codes, option numbers, and content format have been assigned by IANA as Constrained RESTful Environments (CoRE) parameters.**
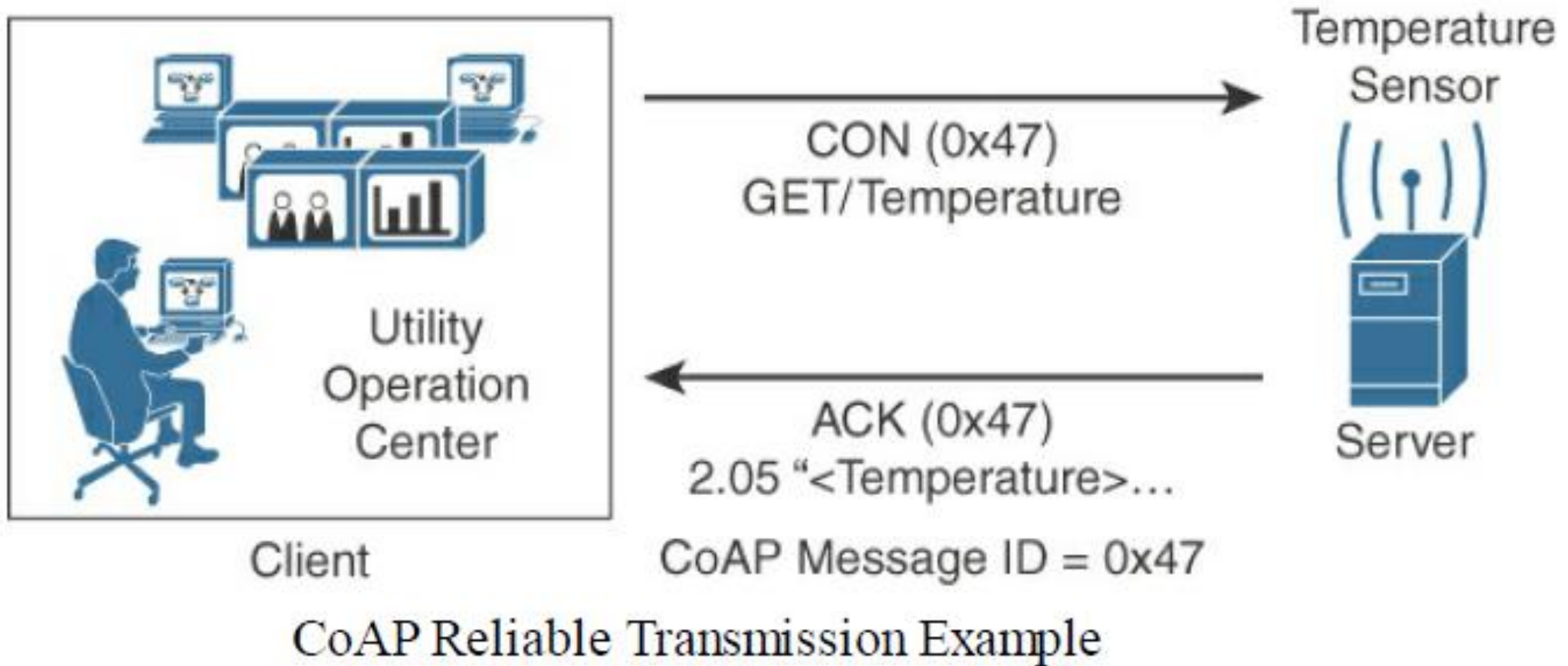
Constrained Application Protocol (CoAP):

➢ **While running over UDP, CoAP offers a reliable transmission of messages when a CoAP header is marked as "confirmable."**

➢ **In addition, CoAP supports basic congestion control with a default time-out, simple stop and wait retransmission with exponential back-off mechanism, and detection of duplicate messages through a message ID.**

➢ **If a request or response is tagged as confirmable, the recipient must explicitly either acknowledge or reject the message, using the same message ID, as shown in Figure.**

➢ **If a recipient can't process a non-confirmable message, a reset message is sent.**

Constrained Application Protocol (CoAP):

CoAP Reliable Transmission Example

Constrained Application Protocol (CoAP):

➢ **Figure shows a utility operations center on the left, acting as the CoAP client, with the CoAP server being a temperature sensor on the right of the figure.**

➢ **The communication between the client and server uses a CoAP message ID of 0x47.**

➢ **The CoAP Message ID ensures reliability and is used to detect duplicate messages.**

➢ **The client sends a GET message to get the temperature from the sensor.**

Constrained Application Protocol (CoAP):

➢ **Notice that the 0x47 message ID is present for this GET message and that the message is also marked with CON.**

➢ **A CON, or confirmable, marking in a CoAP message means the message will be retransmitted until the recipient sends an acknowledgement (or ACK) with the same message ID.**

➢ **temperature sensor does reply with an ACK message referencing the correct message ID of 0x47.**

Constrained Application Protocol (CoAP):

➢ In addition, this ACK message **piggybacks** a successful response to the GET request itself.

➢ This is indicated by the **2.05 response code** followed by the requested data.

➢ CoAP supports data requests sent to a **group of devices** by leveraging the use of IP **Multicast.**

➢ Implementing IP Multicast with CoAP requires the use of **all- CoAP-node multicast** addresses.

➢ For **IPv4** this address is **224.0.1.187**, and for **IPv6 it is FF0X::FD**.

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 typically 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).**

Module – 3 Application Protocols for IoT

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 typically 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).**
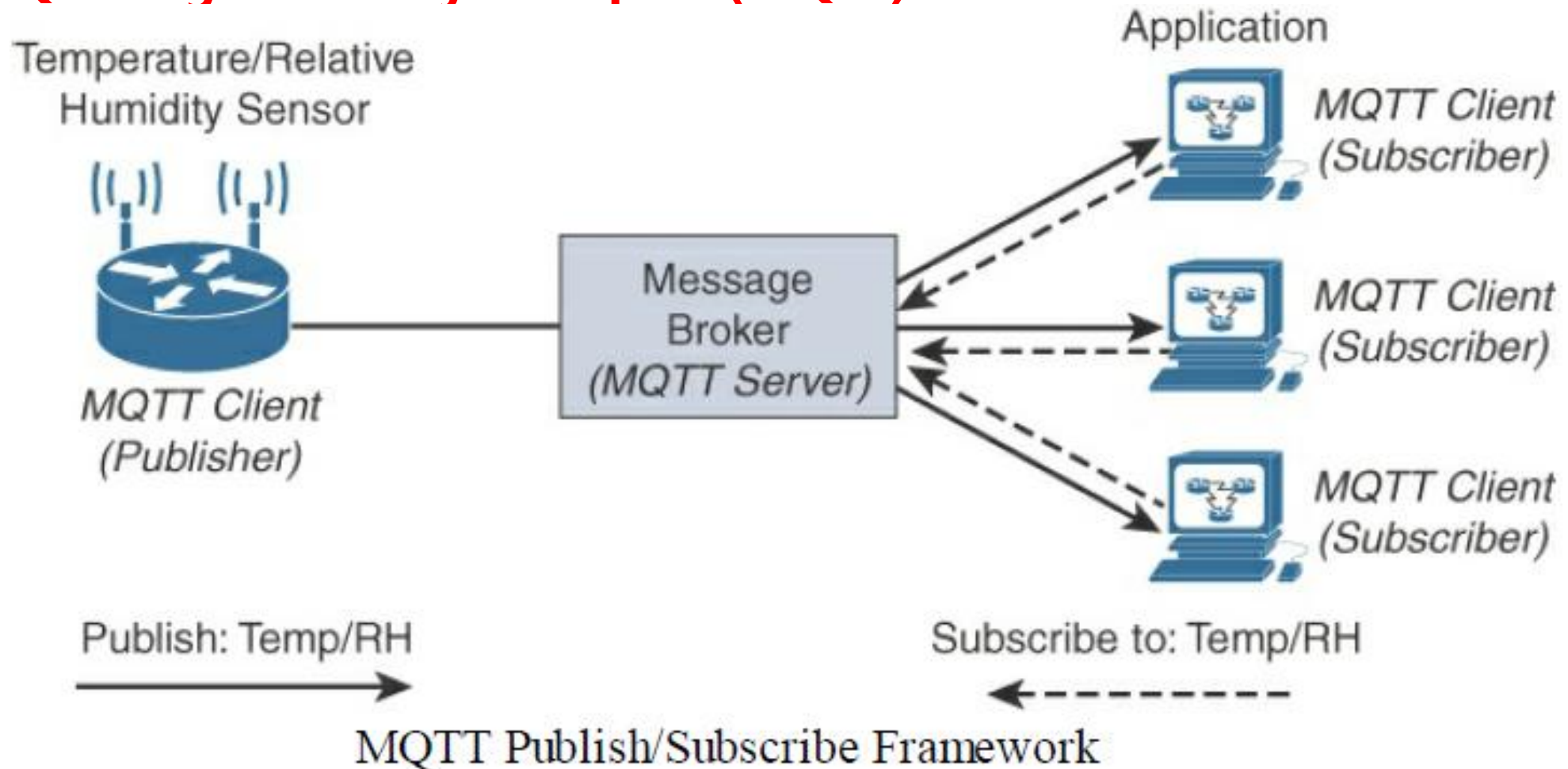
Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **Considering the harsh environments in the oil and gas industries, an extremely simple protocol with only a few options was designed, with considerations for constrained nodes, unreliable WAN backhaul communications, and bandwidth constraints with variable latencies.**

➢ **These were some of the rationales for the selection of a client/server and publish/subscribe framework based on the TCP/IP architecture, as shown in Figure.**

Message Queuing Telemetry Transport (MQTT):

MQTT Publish/Subscribe Framework

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ 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 Figure, the **MQTT client** on the left side is a **temperature** (Temp) and **relative humidity (RH) sensor** that publishes its Temp/RH data.

➢ The MQTT server (or **message broker**) accepts the network connection along with **application messages**, such as **Temp/RH data**, from the **publishers**.

➢ It also handles the **subscription** and **unsubscription** process and pushes the application data to **MQTT clients** acting as **subscribers**.

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **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.**

Message Queuing Telemetry Transport (MQTT):

➢ **The presence of a message broker in MQTT decouples the data transmission between clients acting as publishers and subscribers.**

➢ **In fact, publishers and subscribers do not even know (or need to know) about each other.**

➢ **A benefit of having this decoupling is that the MQTT message broker ensures that information can be buffered and cached in case of network failures.**

➢ **This also means that publishers and subscribers do not have to be online at the same time.**
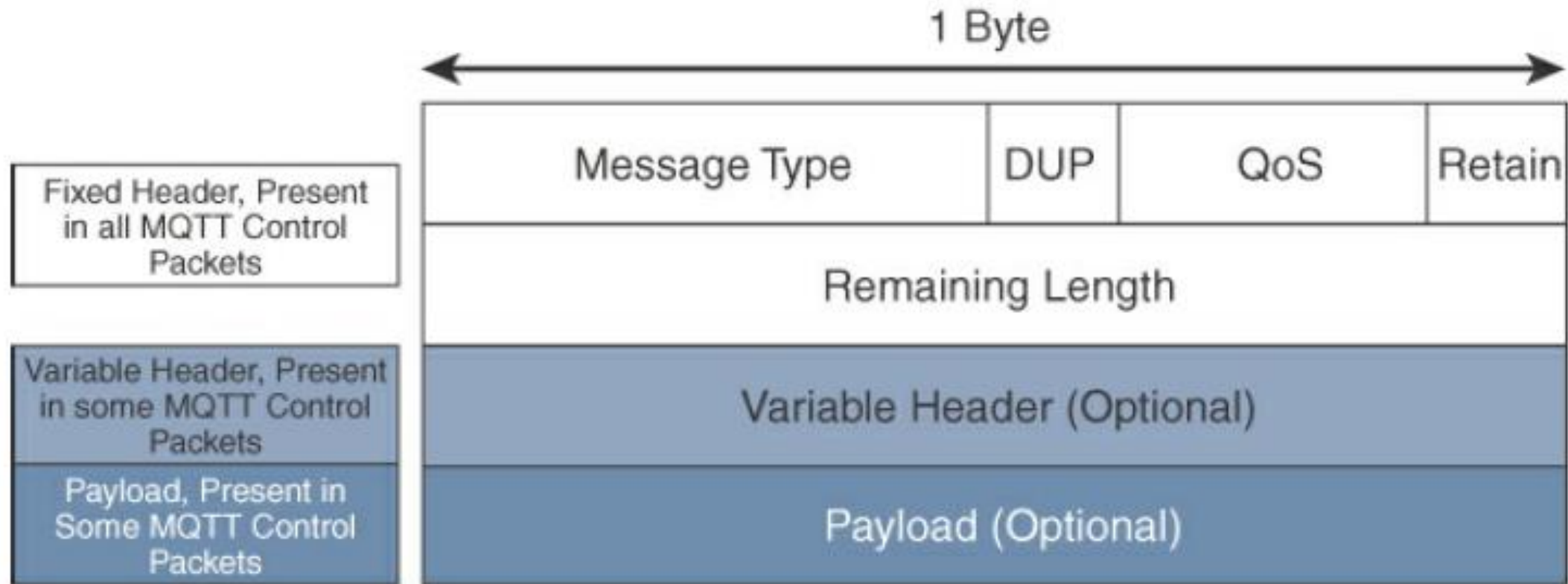
Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➤ **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**.

➤ control packet can contain a payload up to **256 MB**.

Message Queuing Telemetry Transport (MQTT):

Figure provides an overview of the MQTT message format.

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **Compared to the CoAP message format in Figure, MQTT contains a smaller header of 2 bytes compared to 4 bytes for CoAP.**

➢ **The first MQTT field in the header is Message Type, which identifies the kind of MQTT packet within a message.**

➢ **Fourteen different types of control packets are specified in MQTT version 3.1.1.**

➢ **Each of them has a unique value that is coded into the Message Type field.**

Message Queuing Telemetry Transport (MQTT):

MQTT Message Types

Message Type	Value	Flow	Description
CONNECT	1	Client to server	Request to connect
CONNACK	2	Server to client	Connect acknowledgement
PUBLISH	3	Client to server Server to client	Publish message
PUBACK	4	Client to server Server to client	Publish acknowledgement
PUBREC	5	Client to server Server to client	Publish received
PUBREL	6	Client to server Server to client	Publish release

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

MQTT Message Types

Message Type	Value	Flow	Description
PUBCOMP	7	Client to server Server to client	Publish complete
SUBSCRIBE	8	Client to server	Subscribe request
SUBACK	9	Server to client	Subscribe acknowledgement
UNSUBSCRIBE	10	Client to server	Unsubscribe request
UNSUBACK	11	Server to client	Unsubscribe acknowledgement
PINGREQ	12	Client to server	Ping request
PINGRESP	13	Server to client	Ping response
DISCONNECT	14	Client to server	Client disconnecting

Message Queuing Telemetry Transport (MQTT):

➤ **The next field in the MQTT header is DUP (Duplication Flag).**

➤ **This flag, when set, allows the client to notate that the packet has been sent previously, but an acknowledgement was not received.**

➤ **The QoS header field allows for the selection of three different QoS levels.**

➤ **The next field is the Retain flag.**

➤ **Only found in a PUBLISH message , the Retain flag notifies the server to hold onto the message data.**

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **This allows new subscribers to instantly receive the last known value without having to wait for the next update from the publisher.**

➢ **The last mandatory field in the MQTT message header is Remaining Length.**

➢ **This field specifies the number of bytes in the MQTT packet following this field.**

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **MQTT sessions** between each client and server consist of **four phases**:

   Session establishment, authentication, data exchange, and session termination

➢ Each client connecting to a server has a **unique client ID**, which allows the identification of the **MQTT session** between both parties.

➢ When the **server is delivering** an application message to **more than one client**, each client is treated **independently**.

Message Queuing Telemetry Transport (MQTT):

➢ **Subscriptions to resources generate SUBSCRIBE/SUBACK control packets, while unsubscription is performed through the exchange of UNSUBSCRIBE/UNSUBACK control packets.**

➢ **Graceful termination of a connection is done through a DISCONNECT control packet, which also offers the capability for a client to reconnect by re-sending its client ID to resume the operations.**

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

➢ **PINGREQ/PINGRESP control packets are used to validate the connections between the client and server.**

➢ **Similar to ICMP pings that are part of IP, they are a sort of keepalive that helps to maintain and check the TCP session.**

➢ **Securing MQTT connections through TLS is considered optional because it calls for more resources on constrained nodes.**

➢ **When TLS is not used, the client sends a clear-text username and password during the connection initiation.**

Message Queuing Telemetry Transport (MQTT):

- ➢ **The MQTT protocol offers three levels of quality of service (QoS).**

- ➢ **QoS for MQTT is implemented when exchanging application messages with publishers or subscribers.**

- ➢ **The delivery protocol is symmetric.**

- ➢ **This means the client and server can each take the role of either sender or receiver.**

- ➢ **The delivery protocol is concerned solely with the delivery of an application message from a single sender to a single receiver.**

Message Queuing Telemetry Transport (MQTT):

These are the **three levels** of MQTT QoS:

QoS 0:

➢ This is a **best-effort and unacknowledged data service** referred to as "**at most once**" delivery.

➢ The publisher sends its **message one time to a server**, which transmits it once to the subscribers.

➢ **No response** is sent by the receiver, and **no retry** is performed by the sender.

➢ The message arrives at the receiver **either once or not at all**.

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

QoS 1:

➢ **This QoS level ensures that the message delivery between the publisher and server and then between the server and subscribers occurs at least once.**

➢ **In PUBLISH and PUBACK packets, a packet identifier is included in the variable header.**

➢ **If the message is not acknowledged by a PUBACK packet, it is sent again.**

➢ **This level guarantees "at least once" delivery.**

Module – 3 Application Protocols for IoT

Message Queuing Telemetry Transport (MQTT):

QoS 2:

➤ **This is the highest QoS level, used when neither loss nor duplication of messages is acceptable.**

➤ **There is an increased overhead associated with this QoS level because each packet contains an optional variable header with a packet identifier.**

➤ **Confirming the receipt of a PUBLISH message requires a two-step acknowledgement process.**

Message Queuing Telemetry Transport (MQTT):

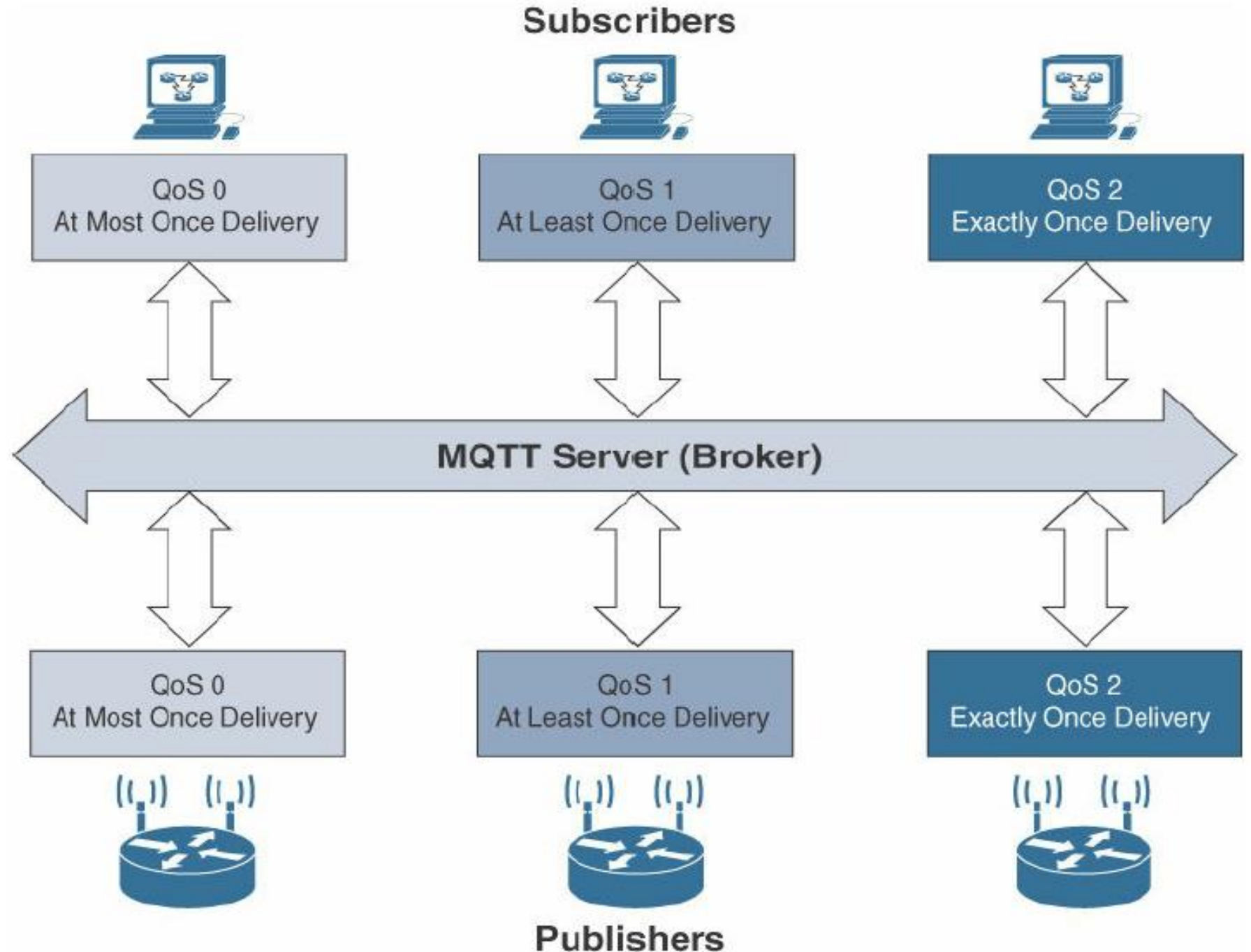**QoS 2:**

➤ **The first step is done through the PUBLISH/PUBREC packet pair, and the second is achieved with the PUBREL/PUBCOMP packet pair.**

➤ **This level provides a "guaranteed service" known as "exactly once" delivery, with no consideration for the number of retries as long as the message is delivered once.**

  • **QoS process is symmetric in regard to the roles of sender and receiver, but two separate transactions exist.**

  • **One transaction occurs between the publishing client and the MQTT server, and the other transaction happens between the MQTT server and the subscribing client.**

Module – 3

Message Queuing

Figure provides an overview of the MQTT QoS flows for the three different levels.

Message Queuing

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
Weaknesses	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

Comparison Between CoAP and MQTT

Message Queuing Telemetry Transport (MQTT):

Summary:

➢ **MQTT is different from the "one-to-one" CoAP model in its "many-to-many" subscription framework**, which can make it a better option for some deployments.

➢ **MQTT is TCP-based**, and it ensures an **ordered and lossless connection**.

➢ **It has a low overhead** when optionally paired with **UDP and flexible message format**, supports **TLS for security**, and provides for **three levels of QoS**.

➢ **This makes MQTT a key application layer protocol** for the successful adoption and **growth of the Internet of Things**.