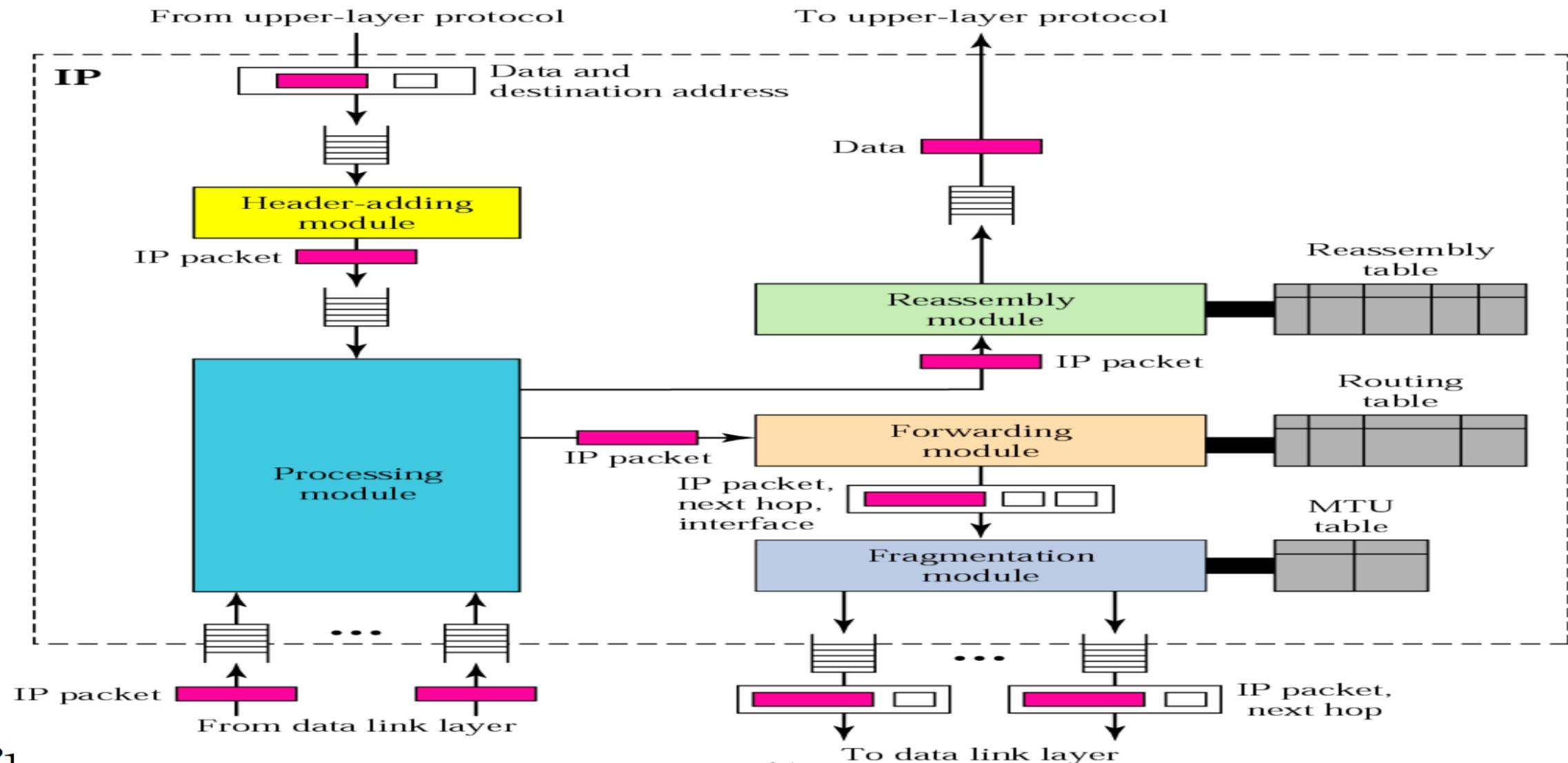


Internet of Things: Addressing & Identification

Lect #10

**Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-769008**

Internet Protocol Design



The IP involves the following components:

- Header-adding module
- Processing module
- Forwarding module
- Reassembly module
- Reassembly table
- Routing table
- MTU table

Header Adding Module

- The Header Adding Module receives data from an upper layer protocol along with the destination IP address.
- It encapsulates the data in an IP datagram by adding the IP header.

Processing Module

- The processing module receives the datagram from an Interface or from the Header-adding module and does the following:
 1. Remove a datagram from one of the input queues.
 2. If the destination address matches loopback address or local address, Send the packet to Reassembly module.
 3. If the machine is a router,
 - decrease TTL.
 4. If TTL is less than or equal to 0,
 - discard the datagram.
 - else send the datagram to the forwarding module

The input queue / The output queue

- The input queues store the datagrams coming from the datalink layer or the upper layer protocols.
- The output queues store the datagrams going to the datalink layer or the upper layer protocols.
- The processing module removes datagrams from the input queues.
- The fragmentation and reassembly modules add the datagram into the output queues.

Forwarding module

- The forwarding module receives an IP packet from the processing module.
- The module finds the IP address of the next-hop along with the interface number to which the packet should be sent.

MTU Table

- The MTU table has two columns Interface number and MTU.
- The MTU table is used by the fragmentation module to find the maximum transfer unit of a particular interface.
- Fragmentation module consults the MTU table to find the MTU of the specific interface number.
- If the length of the datagram is larger than the MTU, the fragmentation module fragments the datagrams, adds a header to each fragment and sends them to the ARP package for address resolution and delivery.

Fragmentation Module

1. Extract the size of the datagram.
2. If size is greater than the MTU of the corresponding network.
3. If “D” bit is set, a discard the datagram. Else, divided the datagram into fragments, add header to each fragment. Add required options to each fragment. Send the datagram.
4. Else send the datagram.

Reassembly Module

- 1.If offset value is 0 and the “more fragment” bit is 0, send the datagram to the appropriate queue.
- 2.Else search the reassembly table for the corresponding entry.(Same D.I. and S.A.)
- 3.If not found, create a new entry.
- 4.Insert the fragment at the appropriate place in the linked list.(based on the offset)
- 5.If all fragments had arrived (“M”-bit and offset), reassemble the fragments, deliver the datagram to the upper layer protocol.
- 6.Check the T.O., if the T.O. expired discard all fragments.

IPV6 Overview

- there is a growing shortage of IPv4 addresses, which are needed for all new devices added to the Internet.
- The key to IPv6 enhancement is the expansion of the IP address space from 32 bits to 128 bits, enabling virtually unlimited, unique IP addresses.
- The new IPv6 address text format is:

xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

Where each x is a hexadecimal digit representing 4 bits.

Simpler IP Configuration

- IPv6 provides new functions that simplify the tasks of configuring and managing the addresses on the network.
- IPv6 reduces some of the workload by automating several of the network administrator's tasks.
- The IPv6 autoconfiguration feature, for example, automatically configures interface addresses and default routes.
- In stateless autoconfiguration, IPv6 takes the Media Access Control (MAC) address of the machine and a network prefix provided by a local router and combines these two addresses to create a new, unique IPv6 address.
- This feature eliminates the need for a Dynamic Host Configuration Protocol (DHCP) server.

IPv6 address formats

- The size and format of the IPv6 address expand addressing capability.
- The IPv6 address size is 128 bits.
- The preferred IPv6 address representation is:
 $x:x:x:x:x:x:x$, Where each x is the hexadecimal values of the eight 16-bit pieces of the address.
- IPv6 addresses range from

0000:0000:0000:0000:0000:0000:0000:0000 to
ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

IPv6 address formats

- In addition to this preferred format, IPv6 addresses might be specified in two other shortened formats:
- **Omit leading zeros**
 - Specify IPv6 addresses by omitting leading zeros. For example, IPv6 address
1050:0000:0000:0000:0005:0600:300c:326b **can be written as**
1050:0:0:0:5:600:300c:326b
- **Double colon**
 - Specify IPv6 addresses by using double colons (:) in place of a series of zeros. For example, IPv6 address
ff06:0:0:0:0:0:c3 can be written as
ff06::c3. Double colons can be used only once in an IP address

Embedding IPV4 into IPV6

- An alternative format for IPv6 addresses combines the colon and dotted notation, so the IPv4 address can be embedded in the IPv6 address.
- Hexadecimal values are specified for the left-most 96 bits, and decimal values are specified for the right-most 32 bits indicating the embedded IPv4 address.
- This format ensures compatibility between IPv6 nodes and IPv4 nodes when you are working in a mixed network environment.

Embedding IPV4 into IPV6

- IPv4-mapped IPv6 address uses this alternative format.
- This type of address is used to represent IPv4 nodes as IPv6 addresses.
- It allows IPv6 applications to communicate directly with IPv4 applications.
- For example, `0:0:0:0:0:ffff:192.1.56.10` and
`::ffff:192.1.56.10 / 96` are the same.

IPv6 address types

- IPv6 addresses are categorized into these basic types:
- **Unicast address**
- **Anycast address**
- **Multicast address**

Unicast address

- The unicast address specifies a single interface.
- A packet sent to a unicast address destination travels from one host to the destination host.
- The two regular types of unicast addresses include:
 - **Link-local address**
 - Link-local addresses are designed for use on a single local link (local network).
 - Link-local addresses are automatically configured on all interfaces.
 - The prefix used for a link-local address is `fe80::/10`.
 - Routers do not forward packets with a destination or source address containing a link-local address.
 - **Global address**
 - Global addresses are designed for use on any network.
 - The prefix used for a global address begins with binary `001`

Special unicast address

- **Unspecified address**

- The unspecified address is `0:0:0:0:0:0:0:0`.
- You can abbreviate the address with two colons (`::`).
- The unspecified address indicates the absence of an address, and it can never be assigned to a host.
- It can be used by an IPv6 host that does not yet have an address assigned to it.
- For example, when the host sends a packet to discover if an address is used by another node, the host uses the unspecified address as its source address.

- **Loopback address**

- The loopback address is `0:0:0:0:0:0:0:1`.
- You can abbreviate the address as `::1`.
- The loopback address is used by a node to send a packet to itself.

Internet of Things: Addressing & Identification

Lect #11

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-769008

IPV6: Anycast Address

- An anycast address specifies a set of interfaces, possibly at different locations, that all share a single address.
- A packet sent to an anycast address goes only to the nearest member of the anycast group.

IPV6: Multicast Address

- The multicast address specifies a set of interfaces, possibly at multiple locations.
- The prefix used for a multicast address is FF.
- If a packet is sent to a multicast address, one copy of the packet is delivered to each member of the group.

IPV6: Neighbour discovery

- Neighbor discovery allows hosts and routers to communicate with one another.
- Neighbor discovery functions are used by IPv6 nodes (hosts or routers) to discover the presence of other IPv6 nodes, to determine the link-layer addresses of nodes, to find routers that are capable of forwarding IPv6 packets, and to maintain a cache of active IPv6 neighbors.

IPV6: Neighbour discovery

- IPv6 nodes use these five Internet Control Message Protocol version 6 (ICMPv6) messages to communicate with other nodes:
- **Router solicitation**
- **Router advertisement**
- **Neighbour solicitation**
- **Neighbour advertisement**
- **Redirect**

IPV6: Neighbour discovery

- **Router solicitation**

- Hosts send these messages to request routers to generate router advertisements.
- A host sends an initial router solicitation when the host first becomes available on the network.

- **Router advertisement**

- Routers send these messages either periodically or in response to a router solicitation.
- The information provided by router advertisements is used by hosts to automatically create global interfaces, and associated routes.
- Router advertisements also contain other configuration information used by a host such as maximum transmission unit and hop limit.

IPV6: Neighbour discovery

- **Neighbour solicitation**
 - Nodes send these messages to determine the link-layer address of a neighbor, or to verify that a neighbor is still reachable.
- **Neighbour advertisement**
 - Nodes send these messages in response to a neighbor solicitation or as an unsolicited message to announce an address change.
- **Redirect**
 - Routers use these messages to inform hosts of a better first hop for a destination.

IPV6 Packet

- An **IPv6 packet** is the smallest message entity exchanged using Internet Protocol version 6 (IPv6).
- Packets consist of control information for addressing and routing and a payload of user data.
- The control information in IPv6 packets is subdivided into a mandatory fixed header and optional extension headers.
- The payload of an IPv6 packet is typically a datagram or segment of the higher-level transport layer protocol, but may be data for an internet layer (e.g., ICMPv6) or link_layer instead.

IPv6 Packet

- IPv6 packets are typically transmitted over the link layer (i.e., over Ethernet or Wi-Fi), which encapsulates each packet in a frame. Packets may also be transported over a higher-layer protocol, such as IPv4 when using 6 to 4 transition technologies.
- In contrast to IPv4, routers do not fragment IPv6 packets larger than the maximum transmission unit (MTU), it is the sole responsibility of the originating node. A minimum MTU of 1,280 octets is mandated by IPv6, but hosts are "strongly recommended" to use Path MTU Discovery to take advantage of MTUs greater than the minimum.

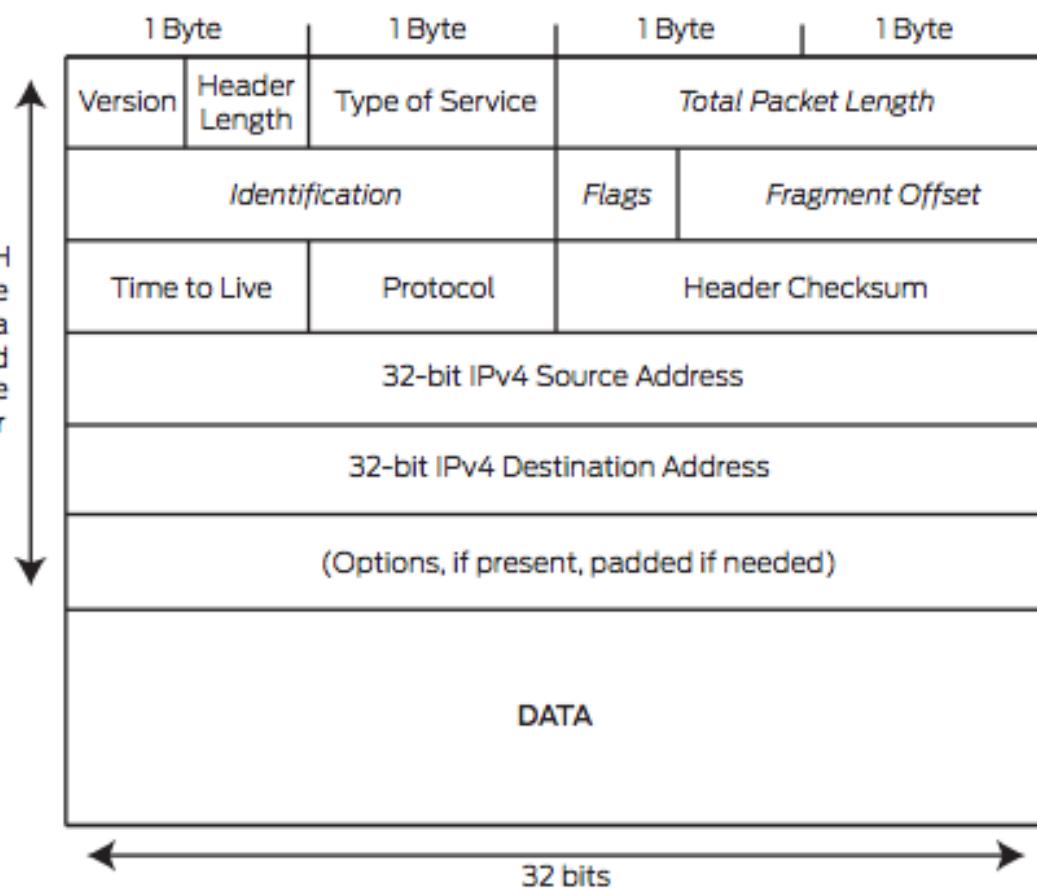
Internet of Things: IPv6 Frame Format

Lect #12

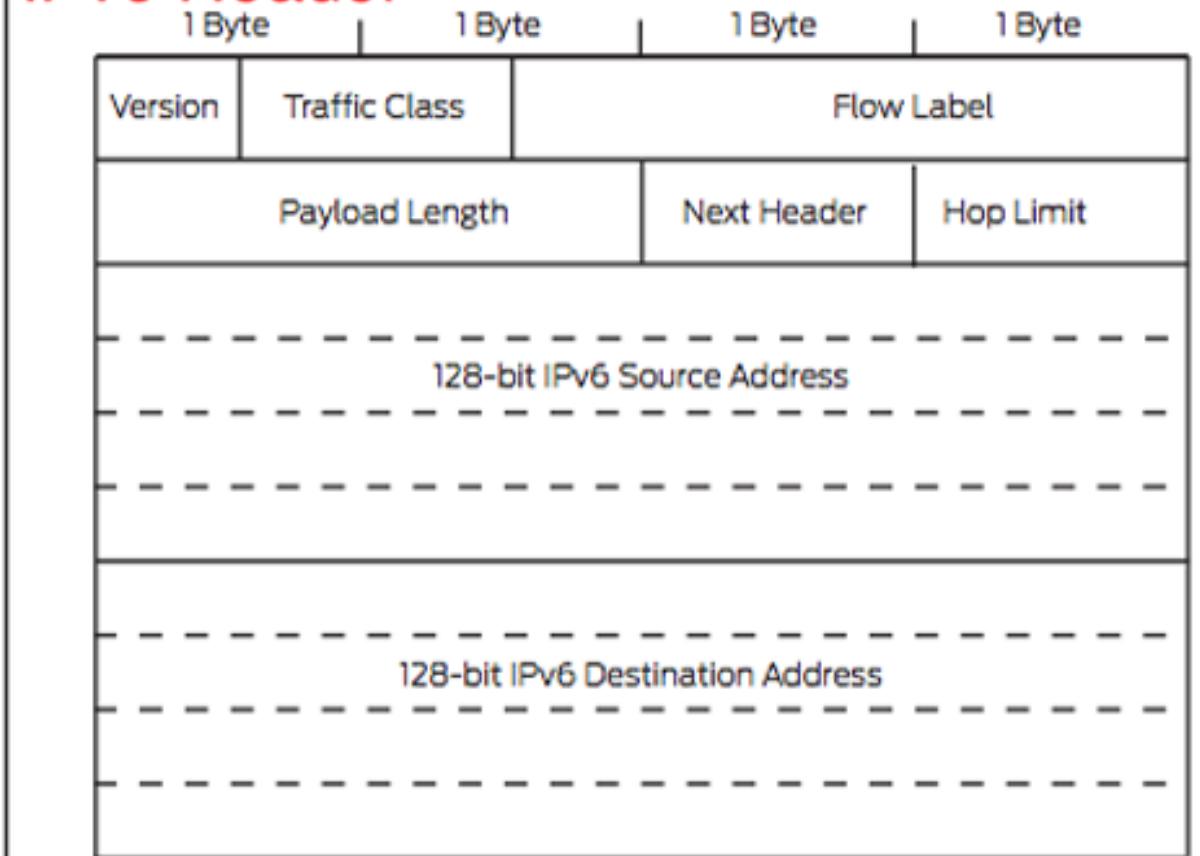
Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-769008

IPv4 vs IPv6 Header

IPv4 Header



IPv6 Header



IPv6 Packet: Fixed header

- The fixed header starts an IPv6 packet and has a size of 40 bytes (octets) / (320 bits)

Fixed header format

Offsets	Octet	0								1								2								3																																	
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31																										
0	0	Version				Traffic class								Flow label																																													
4	32	Payload length																Next header								Hop limit																																	
8	64																																																										
12	96																																																										
16	128																																																										
20	160																																																										
24	192																																																										
28	224																																																										
32	256																																																										
36	288																																																										

IPv6 Packet: Fixed header

- ***Version (4 bits)***
 - The constant 6 (bit sequence 0110)
- ***Traffic Class (6+2 bits)***
 - The bits of this field hold two values. The six most-significant bits hold the differentiated services field (DS field), which is used to classify packets. This replaces ToS field of IPv4 to let the router know what services should be provided to this packet.
 - The remaining two bits are used for Explicit Congestion Notification (ECN); priority values subdivide into ranges: traffic where the source provides congestion control and non-congestion control traffic.

IPv6 Packet: Fixed header

- ***Flow Label (20 bits)***

- This is used to identify the sequence of packets. It helps in prioritizing packet delivery and providing real time service. The vital packets can be delivered ahead of the lower priority packets. The special flow label 0 means the packet does not belong to any flow. Else it identifies flow by source address and port, destination address and port, protocol (value of the last *Next Header* field).

- ***Payload Length (16 bits)***

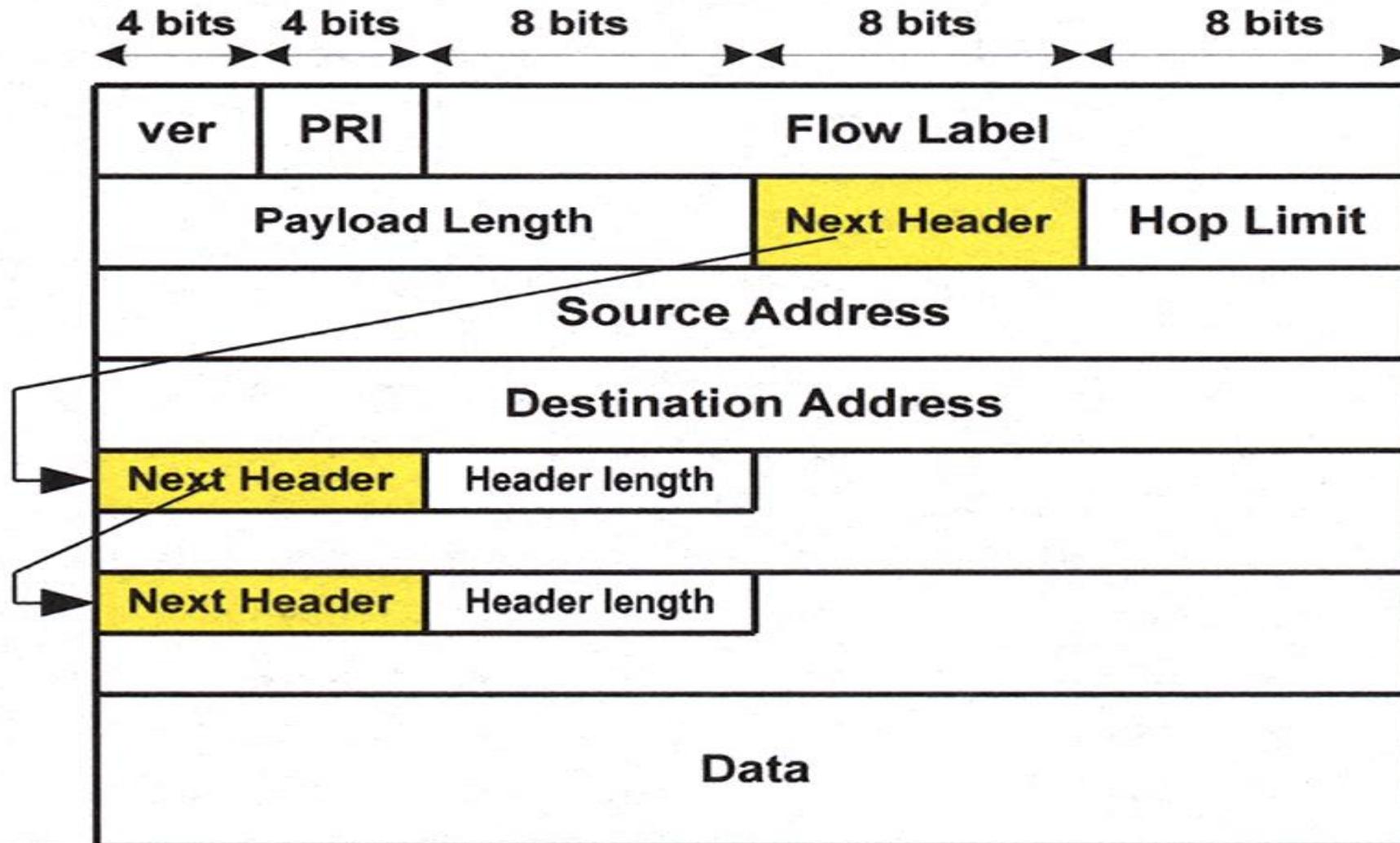
- The size of the payload in bytes, including any extension headers. This identifies the length of the IPv6 payload (the rest of the packet following the IPV6 header). It is used in place of TLEN of IPv4.

IPv6 Packet: Fixed header

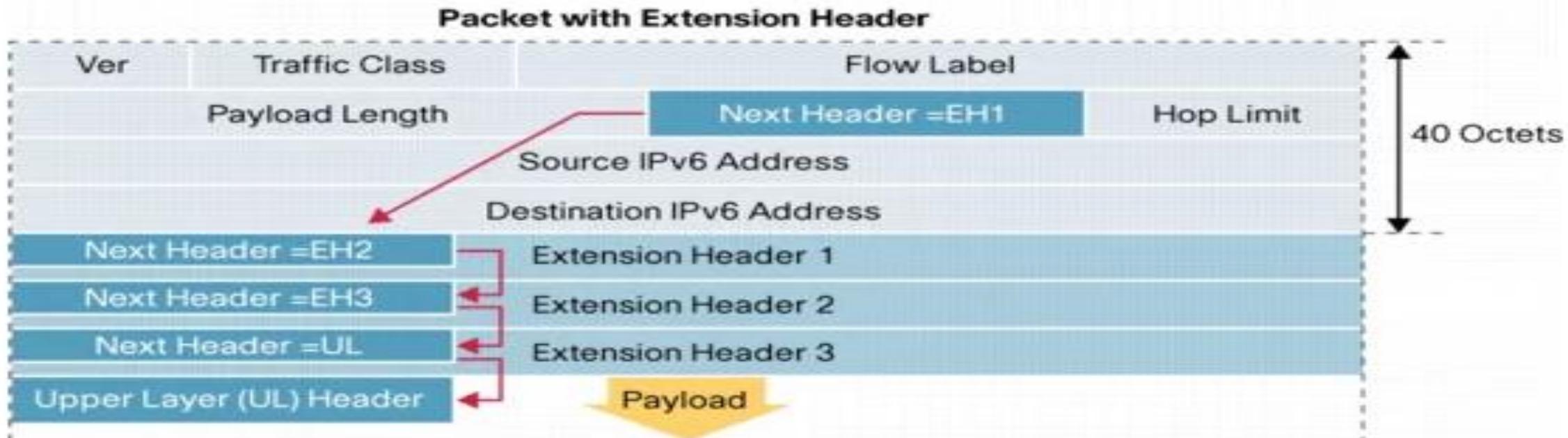
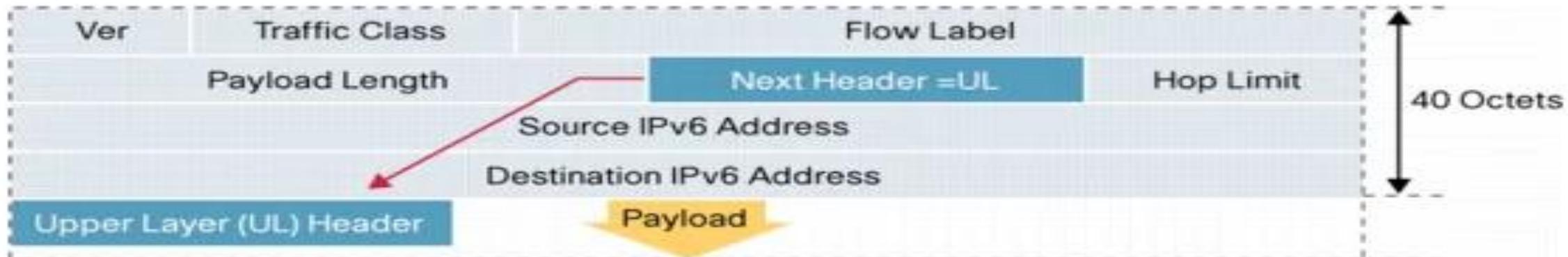
- ***Next Header (8 bits)***

- This is similar to the protocol field in IPv4 header. Specifies the type of the extension header that follows the primary IPv6 header. This field usually specifies the transport layer protocol used by a packet's payload. When extension headers are present in the packet this field indicates which extension header follows.

Next Header (8 bits)



Extension Headers (EH) in IPv6 packets



IPV6 Packet: Fixed header

- ***Hop Limit (8 bits)***

- Replaces the time to live field in IPv4. This value is decremented by one at each forwarding node and the packet is discarded if it becomes 0. However, the destination node should process the packet normally even if received with a hop limit of 0.

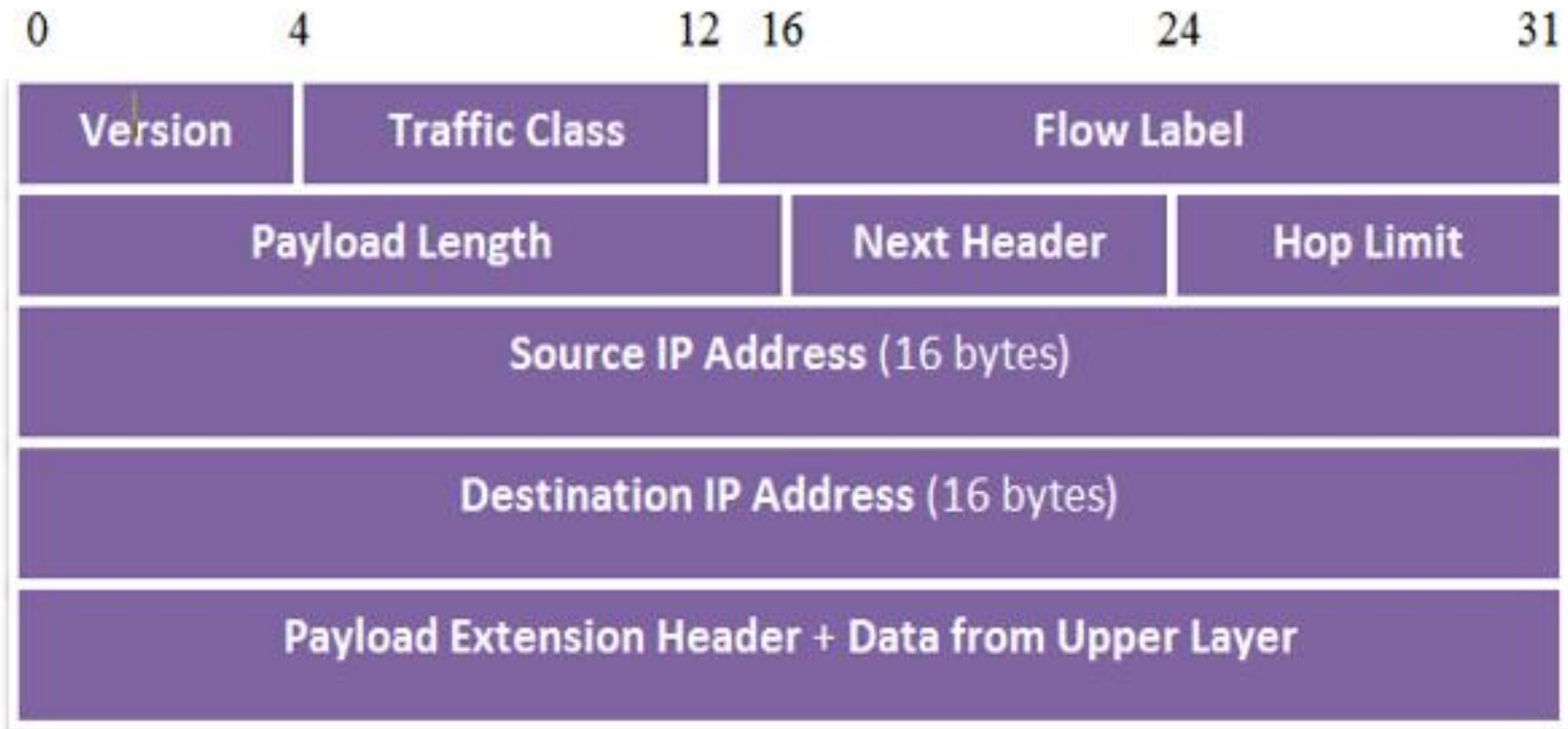
- ***Source Address (128 bits)***

- As in IPv4, this specifies the sender's IP 128-bit address.

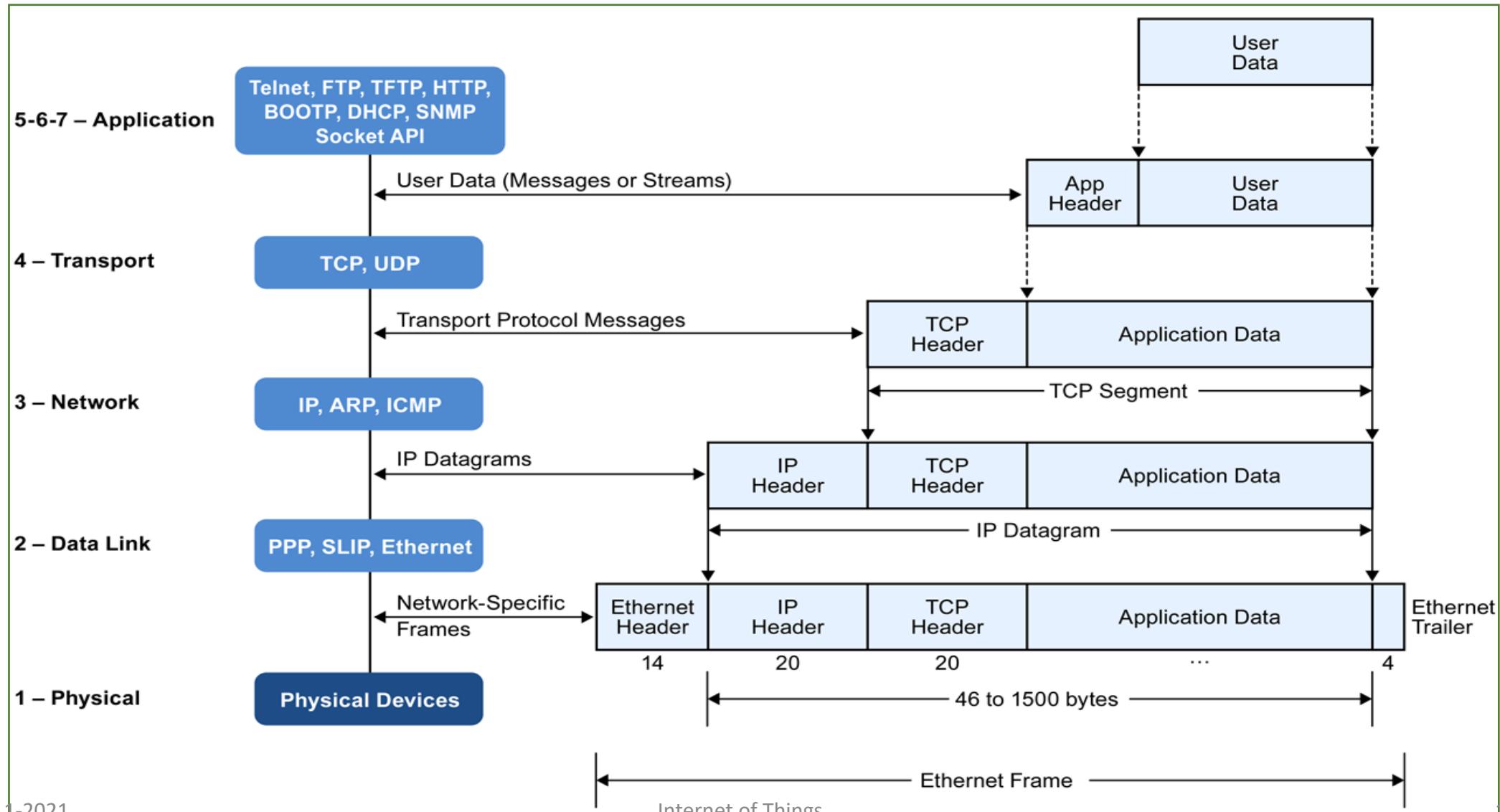
- ***Destination Address (128 bits)***

- Similar to IPv4, the destination's IP address is denoted here.

IPv6 Extension Header



Protocol Stack (Ref)

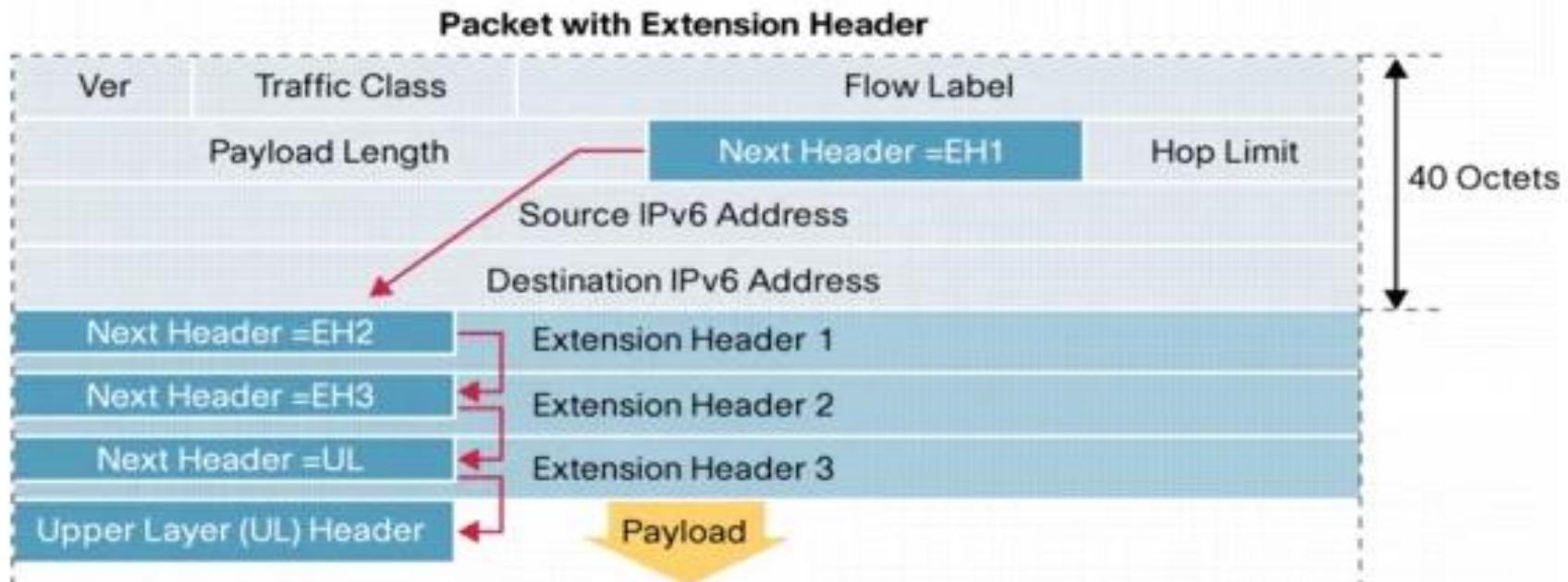
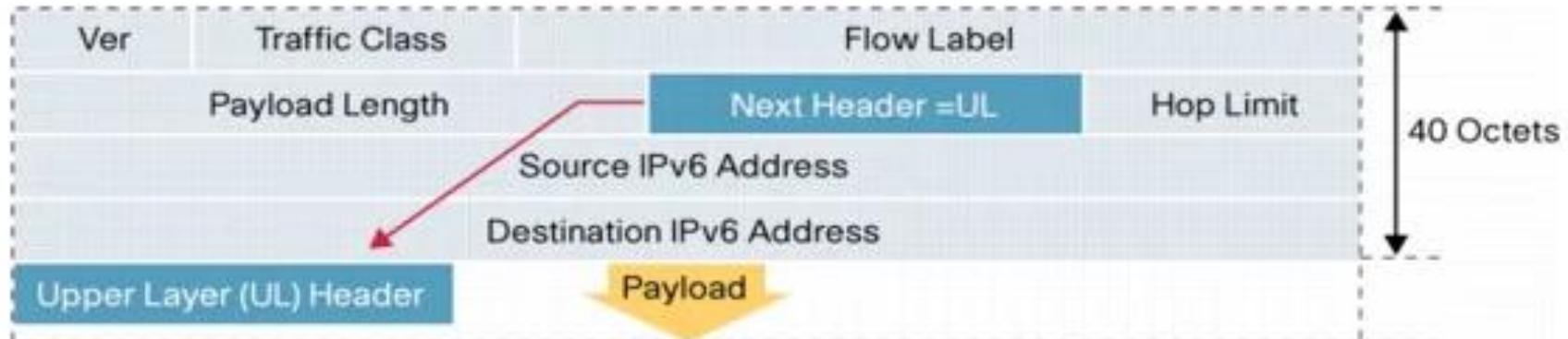


Internet of Things: IPv6 Frame Format, MQTT

Lect #13

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

Extension Headers (EH) in IPv6 packets



IPV6: Extension Header (EH)

- Extension headers (EH) carry optional internet layer information and are placed between the fixed header and the upper-layer protocol header.
- Extension headers form a chain, using the *Next Header* fields.
- The *Next Header* field in the fixed header indicates the type of the first extension header; the *Next Header* field of the last extension header indicates the type of the upper-layer protocol header in the payload of the packet.
- All extension headers are a multiple of 8 octets in size; some extension headers require internal padding to meet this requirement.

IPv6: Extension Header

- There are several extension headers defined, and new extension headers may be defined in the future.
- Most extension headers are examined and processed at the packet's destination.
- *Hop-by-Hop Options* can be processed and modified by intermediate nodes and, if present, must be the first extension (a MUST requirement).
- All extension headers are optional and should only appear at most once, except for the *Destination Options* header extension, which may appear twice.

IPv6: Extension Header

- The defined extension headers below are listed in the preferred order for the case where there is more than one extension header following the fixed header.

IPV6: Extension Headers and Their Recommended Order in a Packet

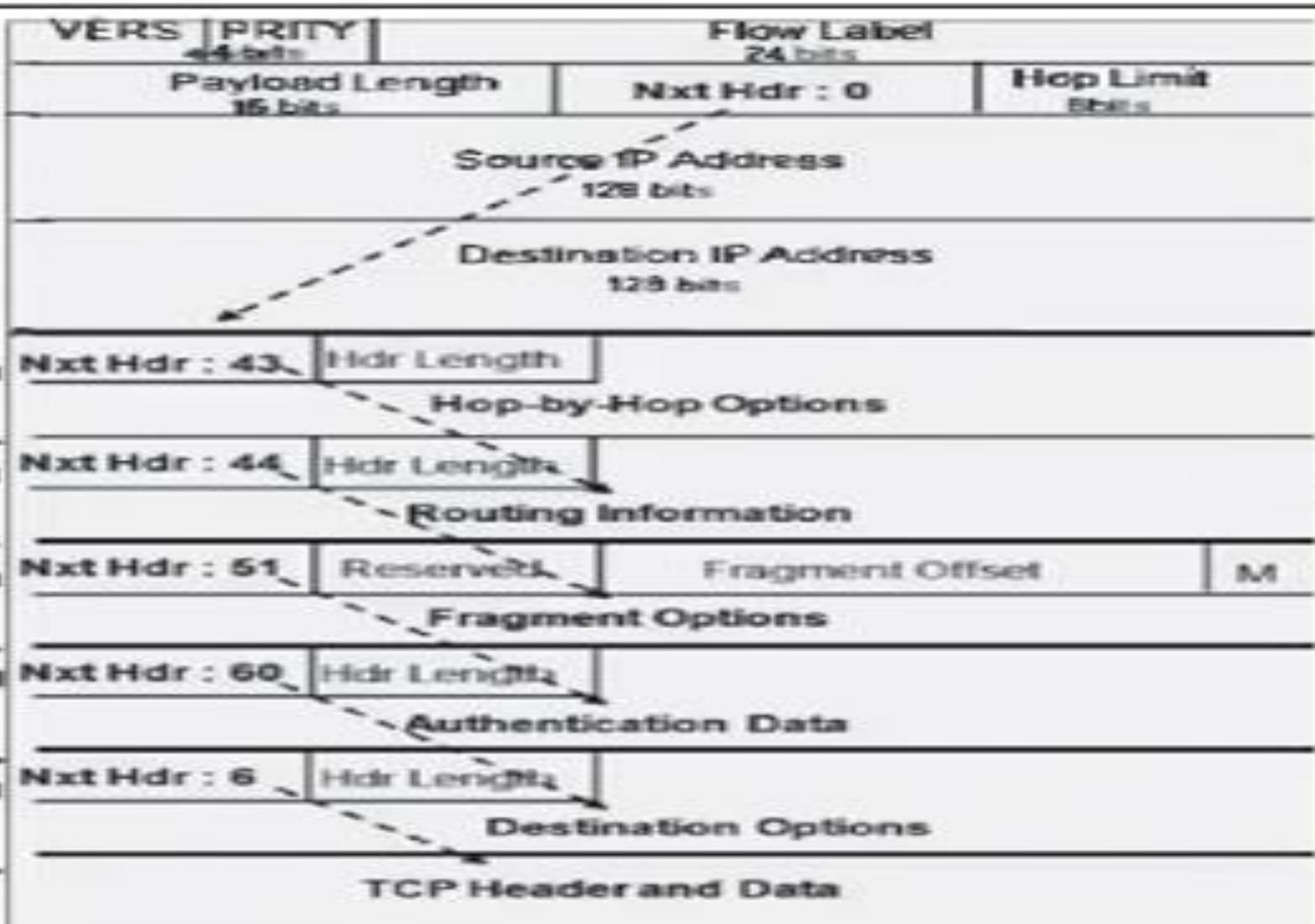
Order	Header Type	Next Header Code
1	Basic IPv6 Header	-
2	Hop-by-Hop Options	0
3	Destination Options (with Routing Options)	60
4	Routing Header	43
5	Fragment Header	44
6	Authentication Header	51
7	Encapsulation Security Payload Header	50
8	Destination Options	60
9	Mobility Header	135
	No next header	59
Upper Layer	TCP	6
Upper Layer	UDP	17
Upper Layer	ICMPv6	58

IPv6 Extension Header

Hop-by-Hop Options and Destination Options extension header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Next header								Header extension length								Options and padding															
4	32									Options and padding																							
8	64																																
12	96									Optional: more Options and padding																							

EHs of IPv6



IPV6: Extension Header

- Value 59 (No Next Header) in the Next Header field indicates that there is no next header *whatsoever* following this one, not even a header of an upper-layer protocol.
- It means that, from the header's point of view, the IPv6 packet ends right after it: the payload should be empty.
- There could, however, still be data in the payload if the payload length in the first header of the packet is greater than the length of all extension headers in the packet.
- This data should be ignored by hosts, but passed unaltered by routers

IPV6 Extension Header: Hop-by-hop options

- The *Hop-by-Hop Options* extension header may be examined and altered by all nodes on the packet's path, including sending and receiving nodes. It is used for the support with the Router Alert option, and is an integral part in the operation of MLD (Multicast Listener Discovery). Router Alert is an integral part in the operations of IPv6 Multicast through MLD and RSVP for IPv6.

IPV6 Extension Header: Destination options

- The *Destination Options* extension header needs to be examined by the destination node(s) only.

IPV6: Extension Header: Routing

- The *Routing* extension header is used to direct a packet to one or more intermediate nodes before being sent to its destination. It is used in IPv6 Mobility and in Source Routing. The header is at least 8 octets in size; if more *Type-specific Data* is needed then it will fit in 4 octets, blocks of 8 octets are added to the header repeatedly, until all *Type-specific Data* is placed.

IPV6 Extension Header: Routing

Routing extension header format

Offsets		Octet	0							1							2							3																												
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31																			
0	0	Next header							Header extension length							Routing type							Segments left																													
4	32	Type-specific data																																																		
8	64	Optional: more type-specific data...																																																		
12	96																																																			

IPV6: Extension Header

- ***Next header* (8 bits)**
 - Indicates the type of the next header.
- ***Header extension length* (8 bits)**
 - The length of this header, in multiples of 8 octets, not including the first 8 octets.
- ***Routing type* (8 bits)**
 - A value between 0 and 255, as assigned by [IANA](#) (**Internet Assigned Numbers Authority**)

IPV6 Extension Header: Routing

- ***Segments Left (8 bits)***
 - Number of nodes this packet still has to visit before reaching its final destination.
- ***Type-specific Data (variable)***

IPV6 Extension Header: Fragment

- In order to send a packet that is larger than the path MTU, the sending node splits the packet into fragments.
- The *Fragment* extension header carries the information necessary to reassemble the original (unfragmented) packet

Fragment extension header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Next header								Reserved								Fragment offset								Res		M					
4	32	Identification																															

IPV6 Extension Header: Fragment

- *Next header* (8 bits)
 - Identifies the type of the next header.
- *Reserved* (8 bits)
 - Initialized to all zeroes.
- *Fragment offset* (13 bits)
 - Offset, in 8-octet units, relative to the start of the fragmentable part of the original packet.
- *Res* (2 bits)
 - Reserved; initialized to zeroes.
- *M Flag* (1 bit)
 - 1 means more fragments follow; 0 means last fragment.
- *Identification* (32 bits)
 - Packet identification value, generated by the source node.
Needed for reassembly of the original packet.

IPV6 Extension Header: Authentication Header (AH) and Encapsulating Security Payload (ESP)

– Same as IPV4

Protocols for IoT- Messaging and Transport

MQTT

Introduction

- Protocols act as the backbone of IoT.
- Messaging protocols play a vital role to send and receive the data/message to /from the cloud for any IoT application.
- Two major protocols in this regard are:
 - MQTT (Message Queuing Telemetry Transport)
 - CoAP (Constrained Application Protocol)

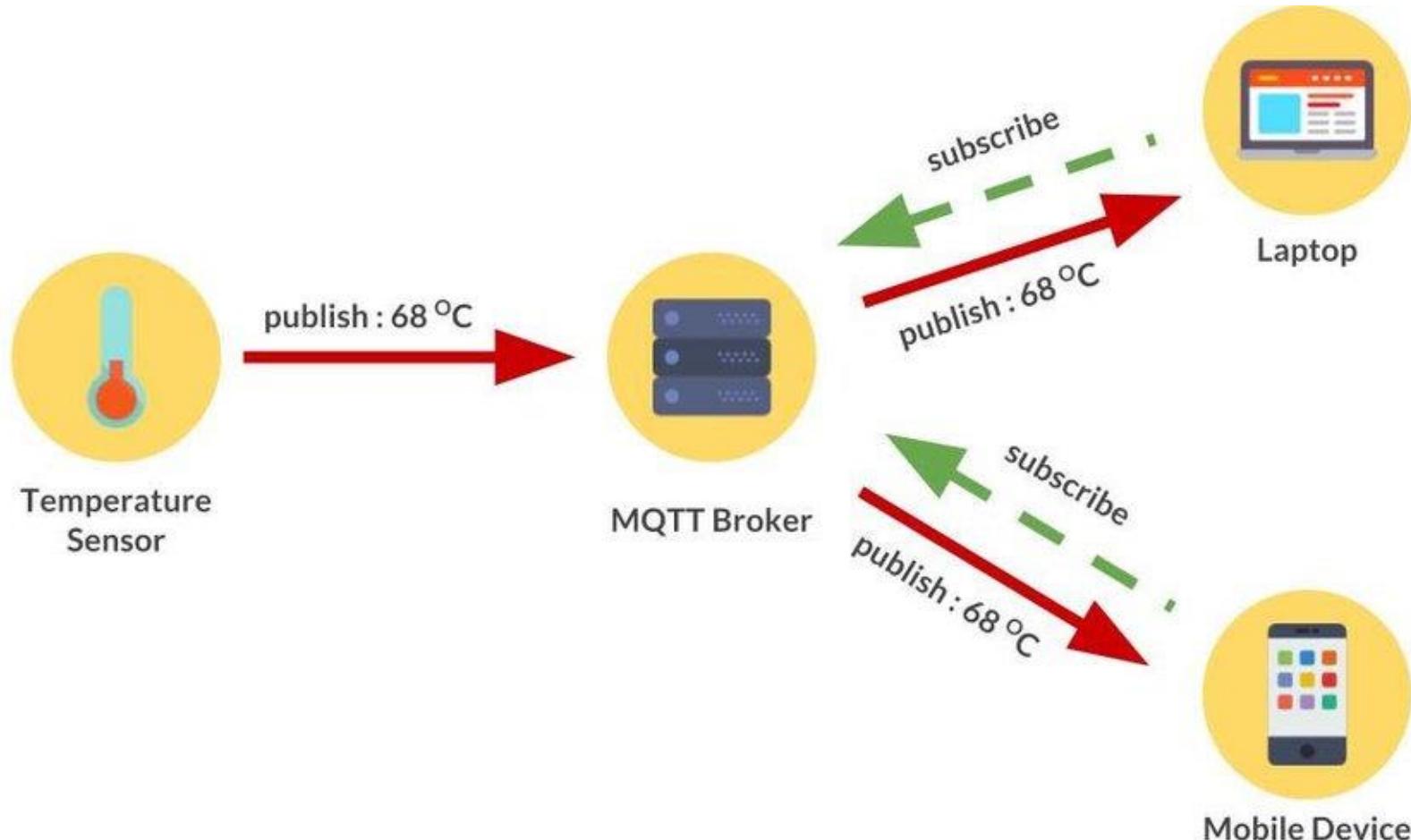
MQTT

- MQTT is a lightweight protocol, which makes it exceptionally useful.
- Lightweight means that it demands minimal resources for its functioning and does not require additional resources from its working environment.
- IoT prefers this type of lightweight protocols due to resource constraint.

MQTT

- MQTT was designed by Andy Stanford Clark & Arlen for IBM 1999 as a proprietary protocol. By 2014, this was made free and open. This began the revolution in IoT.
- MQTT follows the publish – subscribe pattern.
- Thus, a ***central broker*** plays a key role in the entire system.
- This message broker helps to dispatch messages to the nodes that have subscribed for the same.
- The publisher node sends messages to the broker and the broker takes responsibility of dispatching the messages to the subscribed destinations.

Example: MQTT broker



Internet of Things: MQTT (Message Queuing Telemetry Transport)

Lect #14

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

Characteristics of MQTT

- It is a machine to machine protocol. (M2M)
- It is a simple messaging protocol designed for the constrained devices and with low bandwidth
- It provides faster data transmission, like how WhatsApp/messenger provides a faster delivery. It's a real-time messaging protocol.
- The latest version of MQTT (Ver 5.0) has the feature of enhancement in the scalability and the large-scale system in order to set up with the thousands or the millions of devices. (IoT)

MQTT Architecture

- **To understand the MQTT architecture, we first look at the components of the MQTT:**
 - Message
 - Client
 - Server or Broker
 - TOPIC

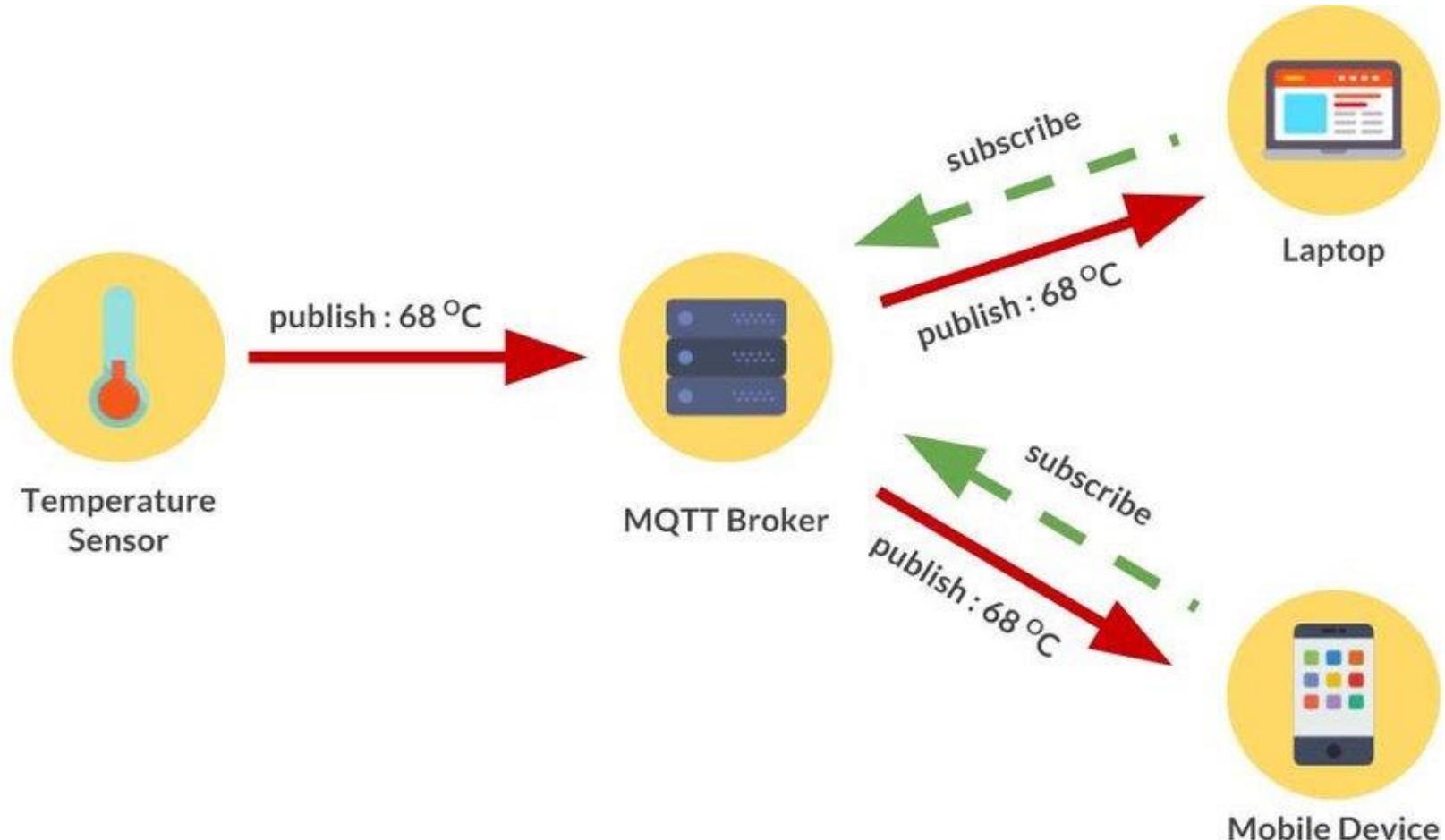
Message

- The message is the data that is carried out by the protocol across the network for the application.
- When the message is transmitted over the network, then the message contains the following parameters:
 - Payload data
 - Quality of Service (QoS) (0 (unAcked)/1(Acked, retransmitted)/2 (RTS, CTS)
 - Collection of Properties
 - Topic Name

Client

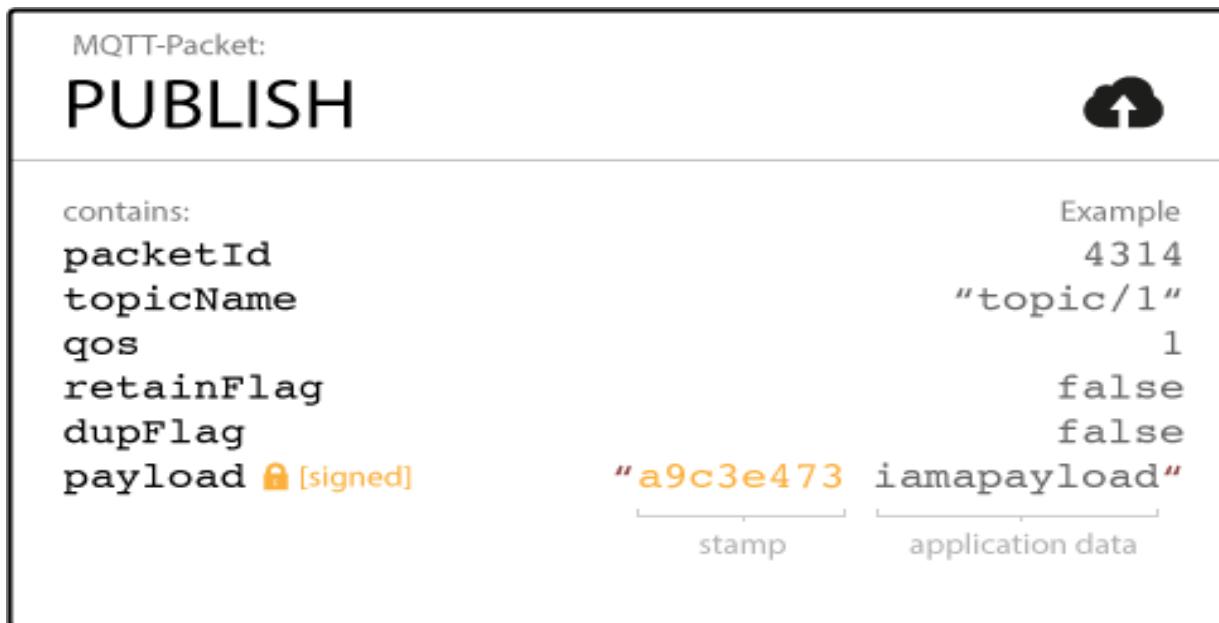
- In MQTT, the subscriber and publisher are the two roles of a client.
- The clients subscribe to the topics to publish and receive messages.
- A device is a client if it opens the network connection to the server, publishes messages that other clients want to see, subscribes to the messages that it is interested in receiving, unsubscribes to the messages that it is not interested in receiving, and closes the network connection to the server. (*If any program or device uses an MQTT, then that device is referred to as a client.*)

Example: Client & Server



Client

- In MQTT, the client performs two operations:
 - **Publish:** When the client sends the data to the server, then we call this operation as a publish.



Client

- **Subscribe:** When the client receives the data from the server, then we call this operation a subscription.



Server / Broker

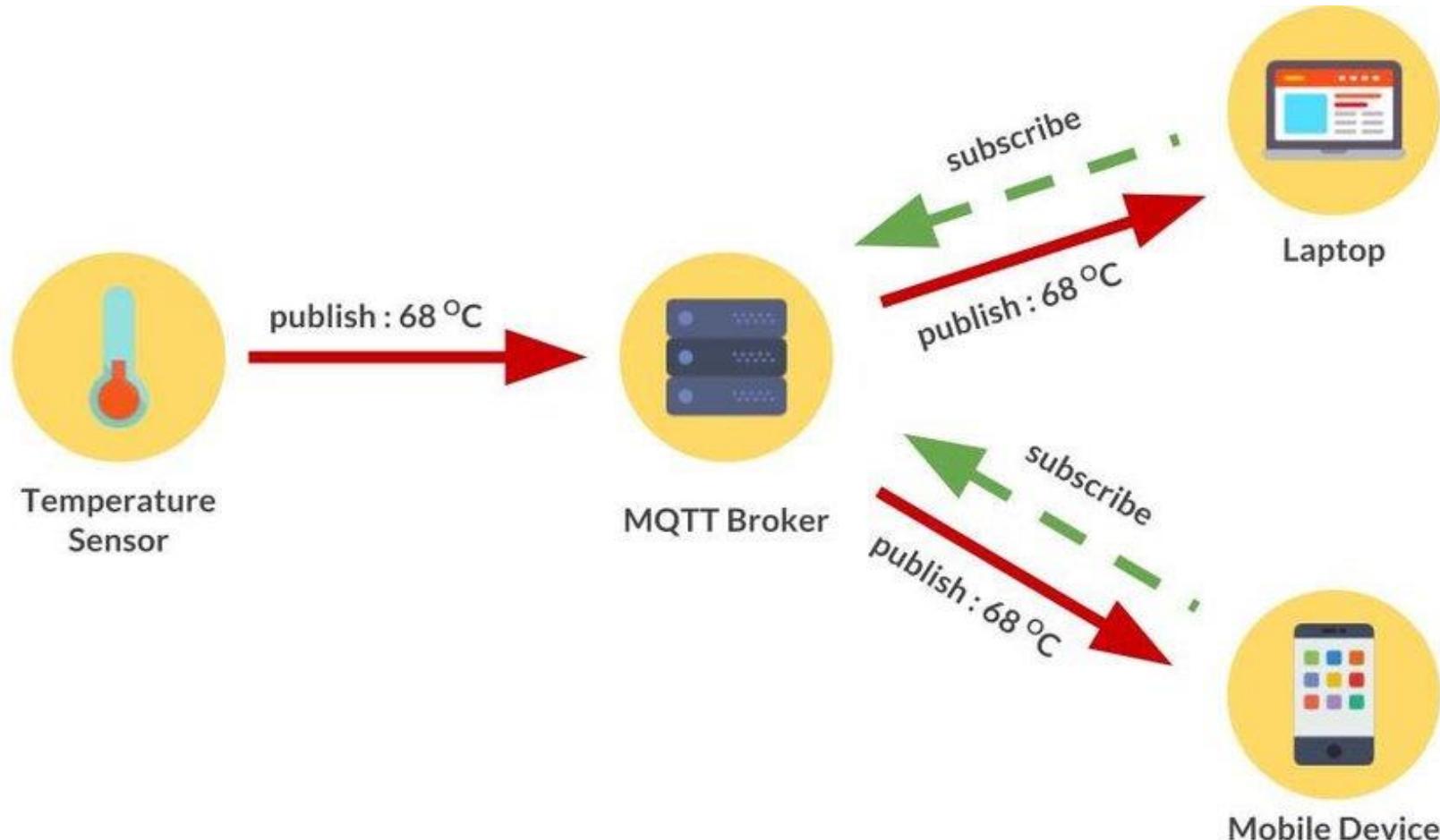
- The device or a program that allows the client to publish the messages and subscribe to the messages.
- A server accepts the network connection from the client, accepts the messages from the client, processes the subscribe and unsubscribe requests, forwards the application messages to the client, and closes the network connection from the client.

TOPIC

- The label provided to the message is checked against the subscription known by the server is known as TOPIC.



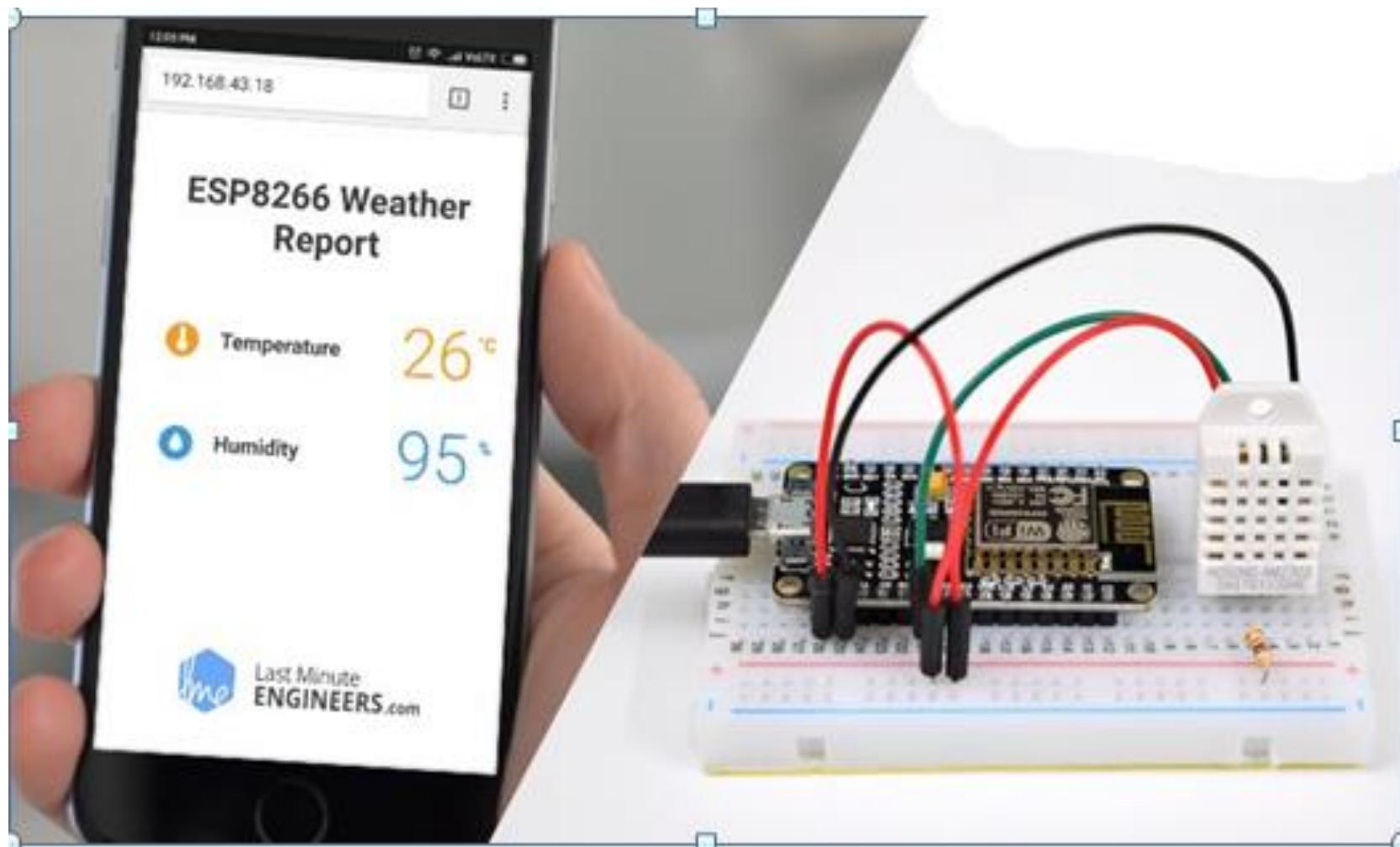
Example: MQTT broker Architecture



Example: MQTT broker Architecture

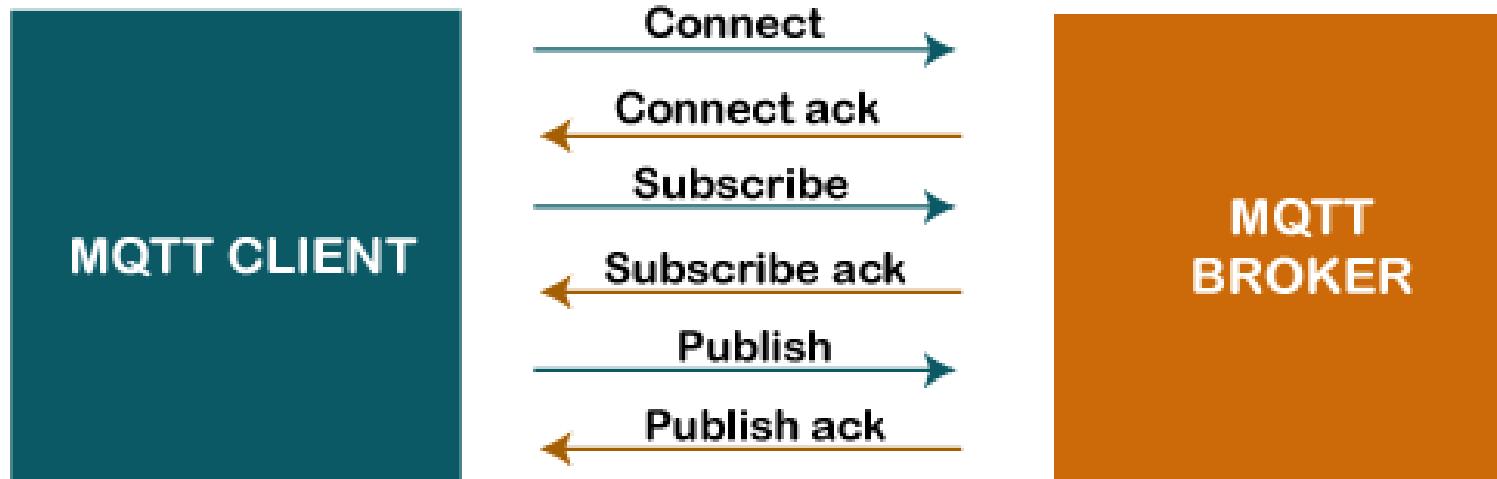
- Suppose a device has a temperature sensor and wants to send the rating to the server or the broker. If the phone or desktop application wishes to receive this temperature value on the other side, then there will be two things that happened.
- The publisher first defines the topic; i.e. temperature and then publishes the message, i.e., the temperature's value. After publishing the message, the phone or the desktop application on the other side will subscribe to the topic, i.e., temperature and then receive the published message, i.e., the value of the temperature.
- The server or the broker's role is to deliver the published message to the phone or the desktop application.

Example: MQTT implementation



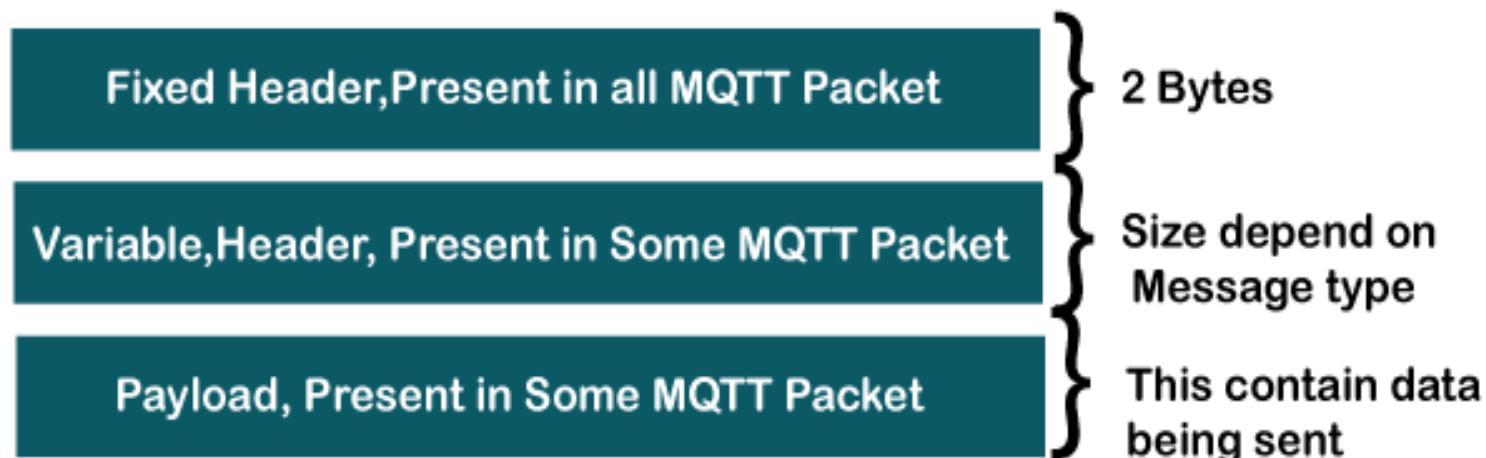
MQTT Message Format

- The MQTT uses the command and the command acknowledgment format.
- This mechanism is similar to the handshaking mechanism as in TCP protocol.

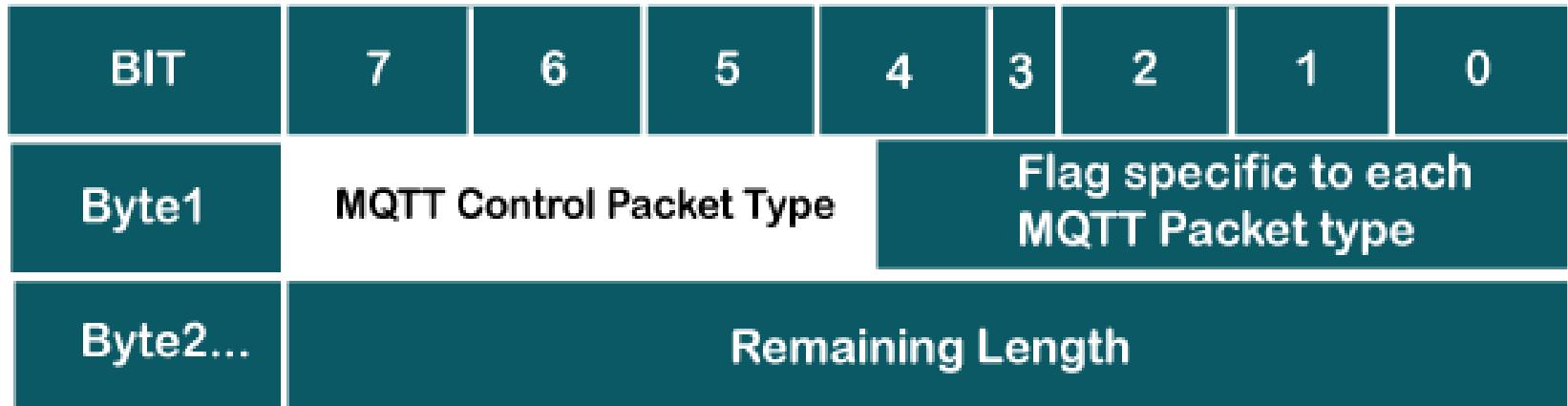


MQTT Packet Structure

- The payload field basically contains the data which is being sent. Some commands do not use the payload field, for example, disconnect message.



Fixed Header -2 bytes



- **MQTT Control Packet Type:** It occupies 4 bits, i.e., 7 to 4-bit positions. This 4-bit is an assigned value, and each bit represents the MQTT control packet type.
- **Flag specific to each MQTT packet type:** The remaining 4-bits represent flag specific to each MQTT packet type.

MQTT Control Packet Types (4 bits)

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

Fixed Header: Remaining Length

- The byte 2 contains the remaining length, which is a variable-length byte integer. It represents the number of bytes remaining in a current control packet, including data in the variable header and payload.
- Remaining length = length of variable header + length of payload
- For example, if the length of the variable header is 20 and the length of the payload is 30, then the remaining length is 50.
- The remaining length starts from 2 bytes and can be upto 4 bytes.
- This field uses 7-bit for the lengths, and MSB bit can be used to continue a flag. If the continuation flag is 1, the next byte is also a part of the remaining length.
- If the continuation flag is 0, the byte is the last one of the remaining length.

Variable Header

- Some types of MQTT control packet types contain an optional field also, i.e., variable header component. This field resides between the fixed header and the payload. The content of the variable header depends upon the packet type. The variable header contains the packet identifier field, which is common in several packet types. The variable header component of many MQTT control packet types includes 2-byte integer, i.e., the packet identifier field.

Example: Packet identifier field

- PUBLISH
- PUBACK
- PUBREC
- PUBREL
- PUBCOMP
- SUBSCRIBE
- SUBACK
- UNSUBSCRIBE
- UNSUBACK

Key points related to the packet identifier field:

- A PUBLISH packet should not contain the packet identifier field if the value of QoS is set to zero. It implies that if the value of QoS is greater than zero, only the PUBLISH packet will contain the packet identifier field.
- When a client sends a new SUBSCRIBE, UNSUBSCRIBE, or PUBLISH MQTT control packet, it should assign a non-zero packet identifier that is currently unused.
- When a server sends a new PUBLISH MQTT control packet, it should assign a non-zero packet identifier that is currently unused.
- A PUBACK, PUBREC, PUBREL, are the acknowledgment packets of PUBLISH command that contain the same packet identifier as the PUBLISH packet.
- A SUBACK and UNSUBACK are the acknowledgment packets of SUBSCRIBE and UNSUBSCRIBE, respectively. Both the packets, i.e., SUBACK and UNSUBACK, use the same packet identifier as the SUBSCRIBE and UNSUBSCRIBE packets.

Payload

- The last MQTT control packet is the payload. This field contains the data which is to be sent. For example, in the case of the CONNECT packet, the payload is a client ID, the username and password, and the PUBLISH packet, the payload is just an application message.

MQTT control Packet that contain Payload

MQTT Control Packet	Payload
CONNECT	Required
CONNACK	None
PUBLISH	Optional
PUBACK	None
PUBREC	None
PUBREL	None
PUBCOMP	None
SUBSCRIBE	Required
SUBACK	Required
UNSUBSCRIBE	Required
UNSUBACK	Required
PINGREQ	None
PINGRESP	None
DISCONNECT	None
AUTH	None

Internet of Things: CoAP (Constrained Application Protocol)

Lect #15

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

CoAP Introduction

- CoAP is a simple protocol with low overhead specifically designed for constrained devices (such as microcontrollers) and constrained networks.
- This protocol is used in M2M data exchange.
- CoAP is a one-to-one communication protocol.
- It is used to connect the smart devices to internet.
- CoAP is a lightweight protocol.

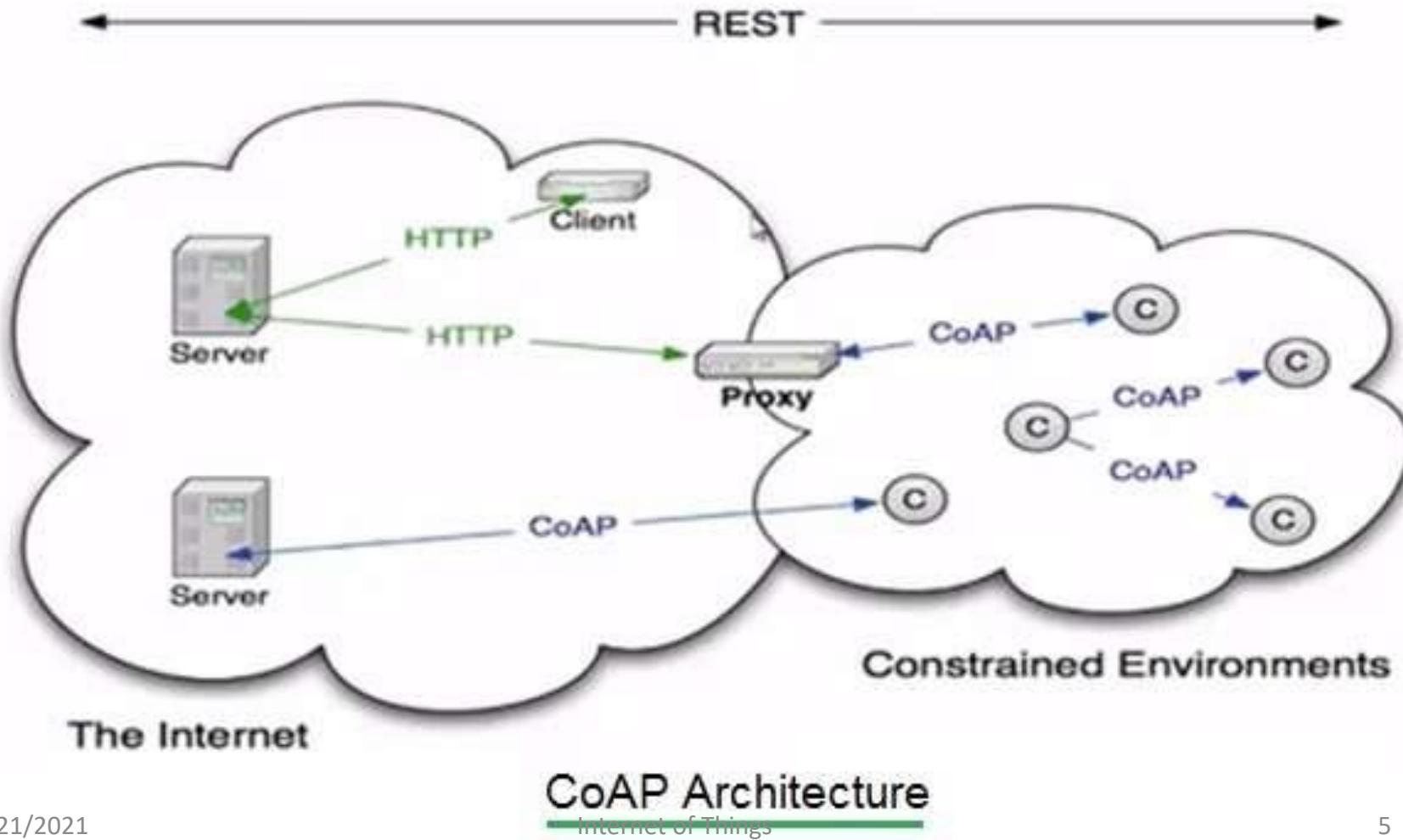
CoAP Vs HTTP

- some features are very similar to HTTP
- HTTP is highly favored and has no constraints or restrictions.
- CoAP is specially designed for constrained environments.
- HTTP runs over TCP and connection oriented.
- CoAP runs over UDP and is connectionless.

CoAP

- Client- server model is the idea behind CoAP.
- As per the model, the client sends a request followed by server responses to the request with appropriate reply or an error message.

CoAP architecture

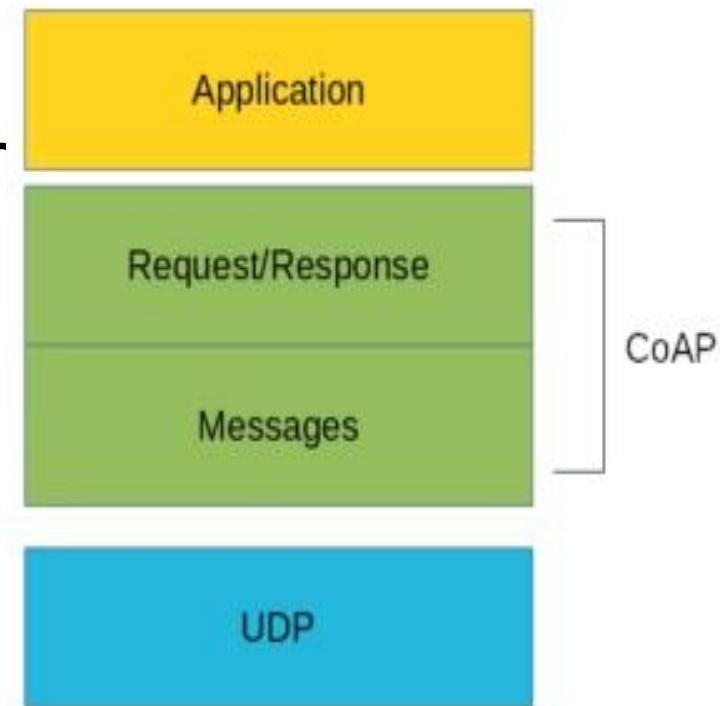


CoAP architecture

- CoAP is based on the REST (Representational State Transfer) API model, known as RESTful.
- REST approach ensures a secure, fault-tolerant and scalable system.
- Since CoAP is based on UDP and is a connectionless protocol, it requires retransmission support.

Layers of CoAP

- There are two different layers that make CoAP protocol: Messages and Request/Response.
- The Messages layer deals with UDP and with asynchronous messages.
- The Request/Response layer manages request/response interaction based on request/response messages.



Messaging components

- **Endpoint:** An entity that participates in the CoAP protocol. Usually, an Endpoint is identified with a host.
- **Sender:** The entity that sends a message
- **Recipient:** The destination of a message
- **Client:** The entity that sends a request and also the destination of the response
- **Server:** The entity that receives a request from a client and sends back a response to the client

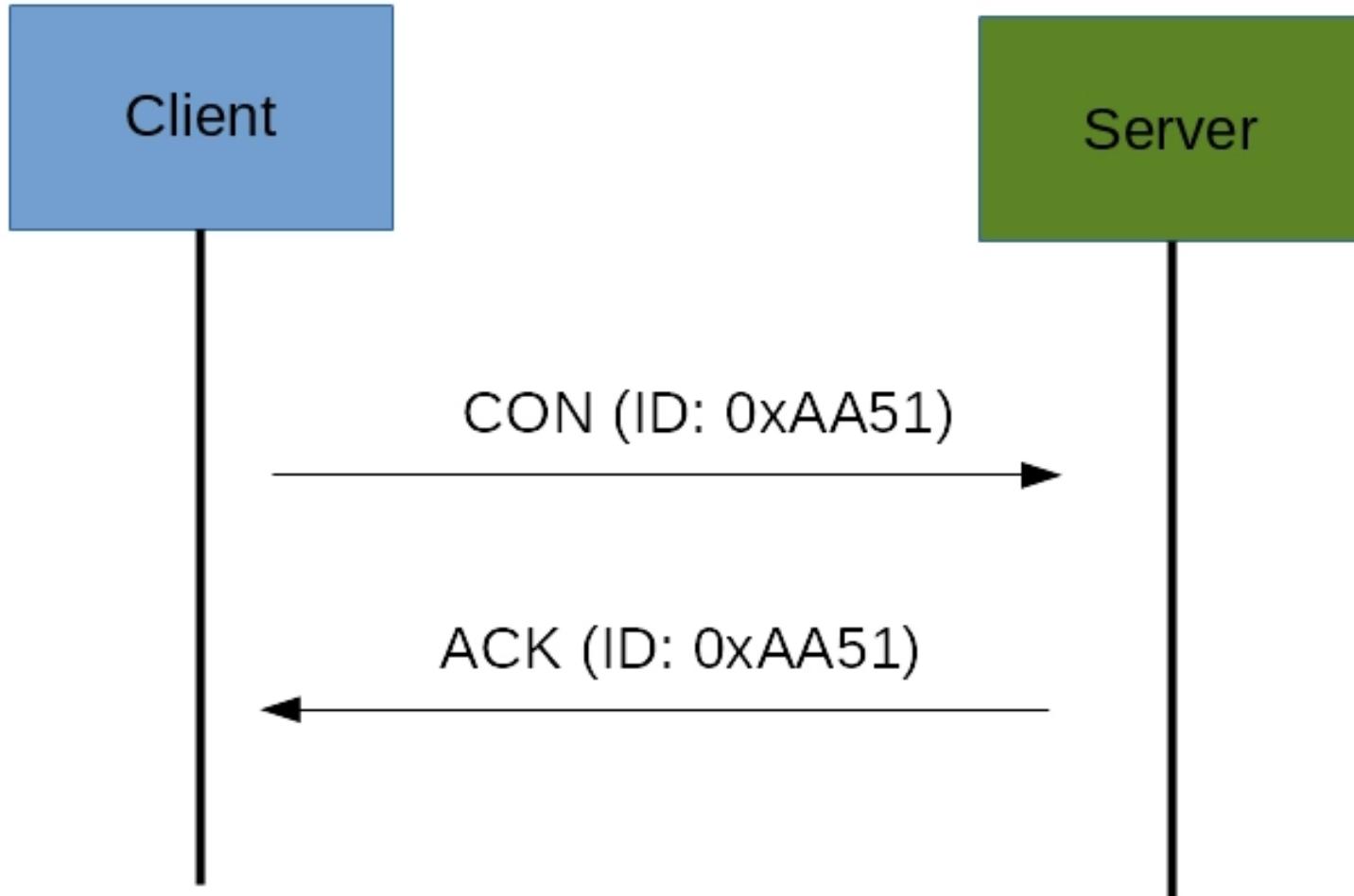
CoAP Messages

- CoAP supports four different message types:
 - Confirmable
 - Non-confirmable
 - Acknowledgment
 - Reset

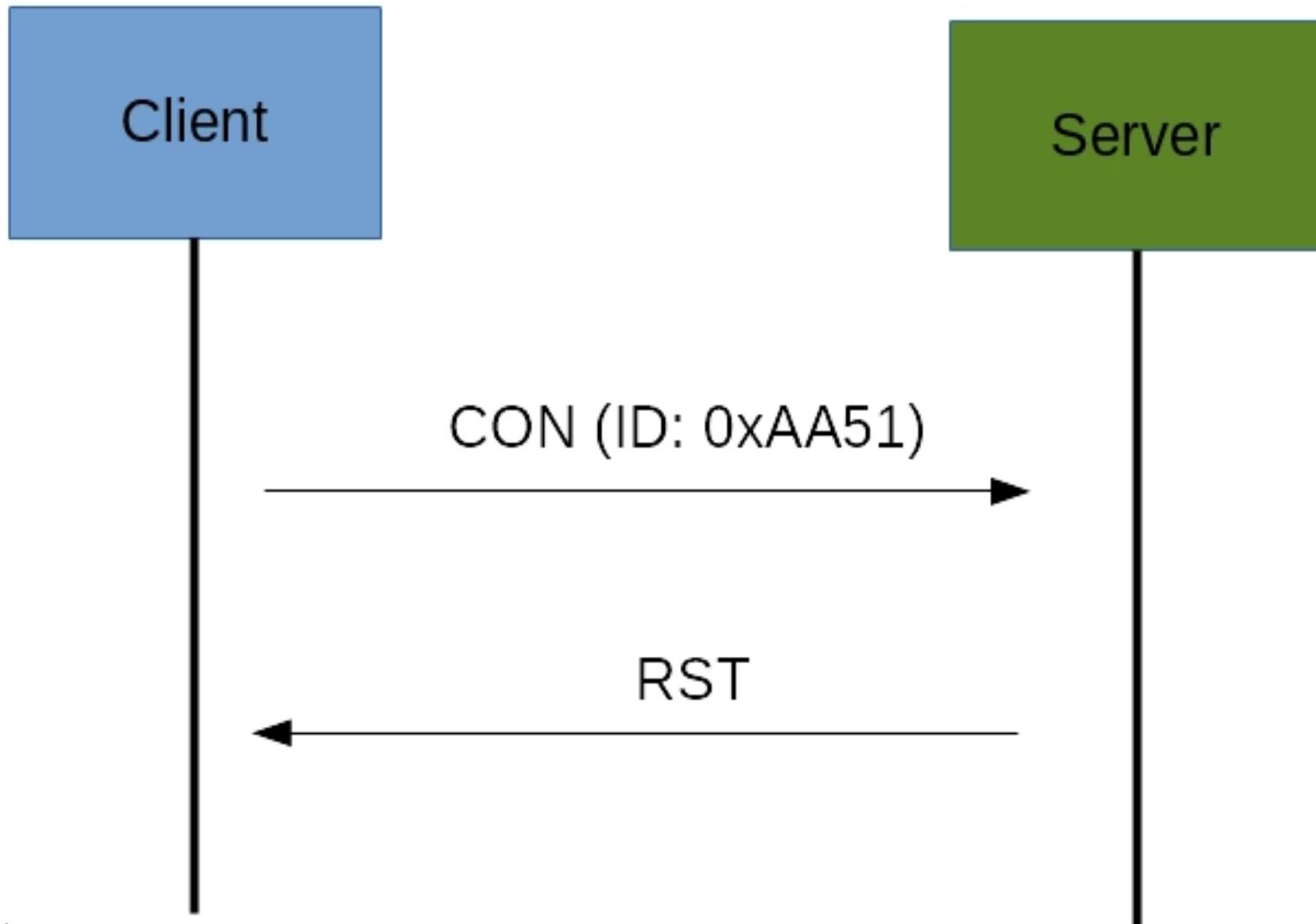
Confirmable Message (Reliable)

- A confirmable message is a reliable message.
- In CoAP, a reliable message is obtained using a Confirmable message (CON).
- Using this kind of message, the client can be sure that the message will arrive at the server.
- A Confirmable message is sent again and again until the other party sends an acknowledge message (ACK).
- The ACK message contains the same ID of the confirmable message (CON).
- When there is a timeout or fail, there would be a RST message sent from the server as response.

Example: Confirmable Message



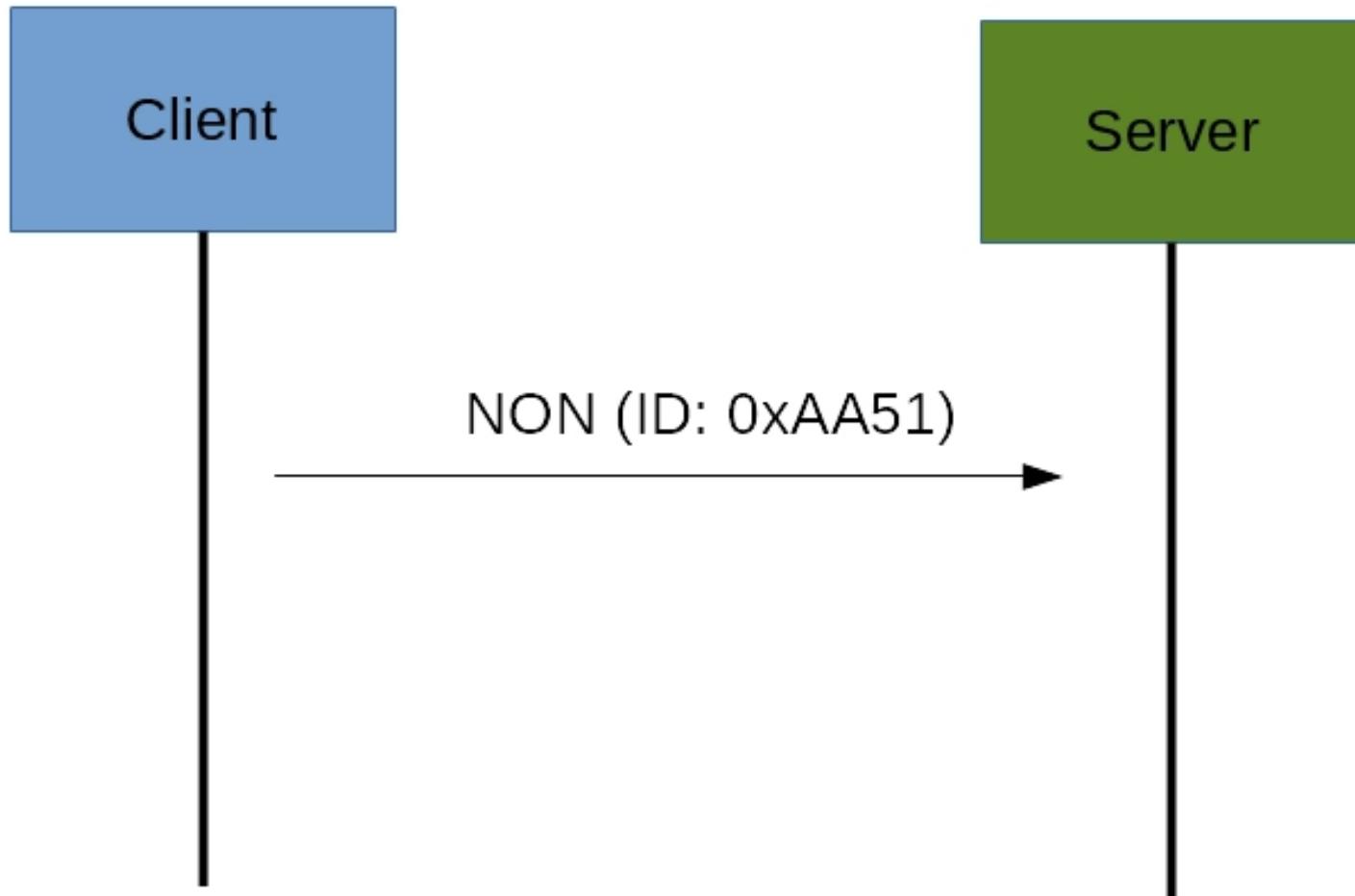
Example: Confirmable Message



Non Confirmable Message (Non-reliable)

- It is an unreliable message transmission style.
- These are messages that don't require an Acknowledgement by the server.
- They are unreliable messages or in other words messages that do not contain critical information that must be delivered to the server.
- To this category belongs messages that contain values read from sensors.

Example: Non conformable message



Acknowledgement

- It's a traditional acknowledgement message.
- Used in handshaking scheme for connection establishment
- Ensures reliability in message exchanging

RST

- If no processing has happened at the receiver end even after a specific amount of time, there should be a mechanism to inform the sender of the situation.

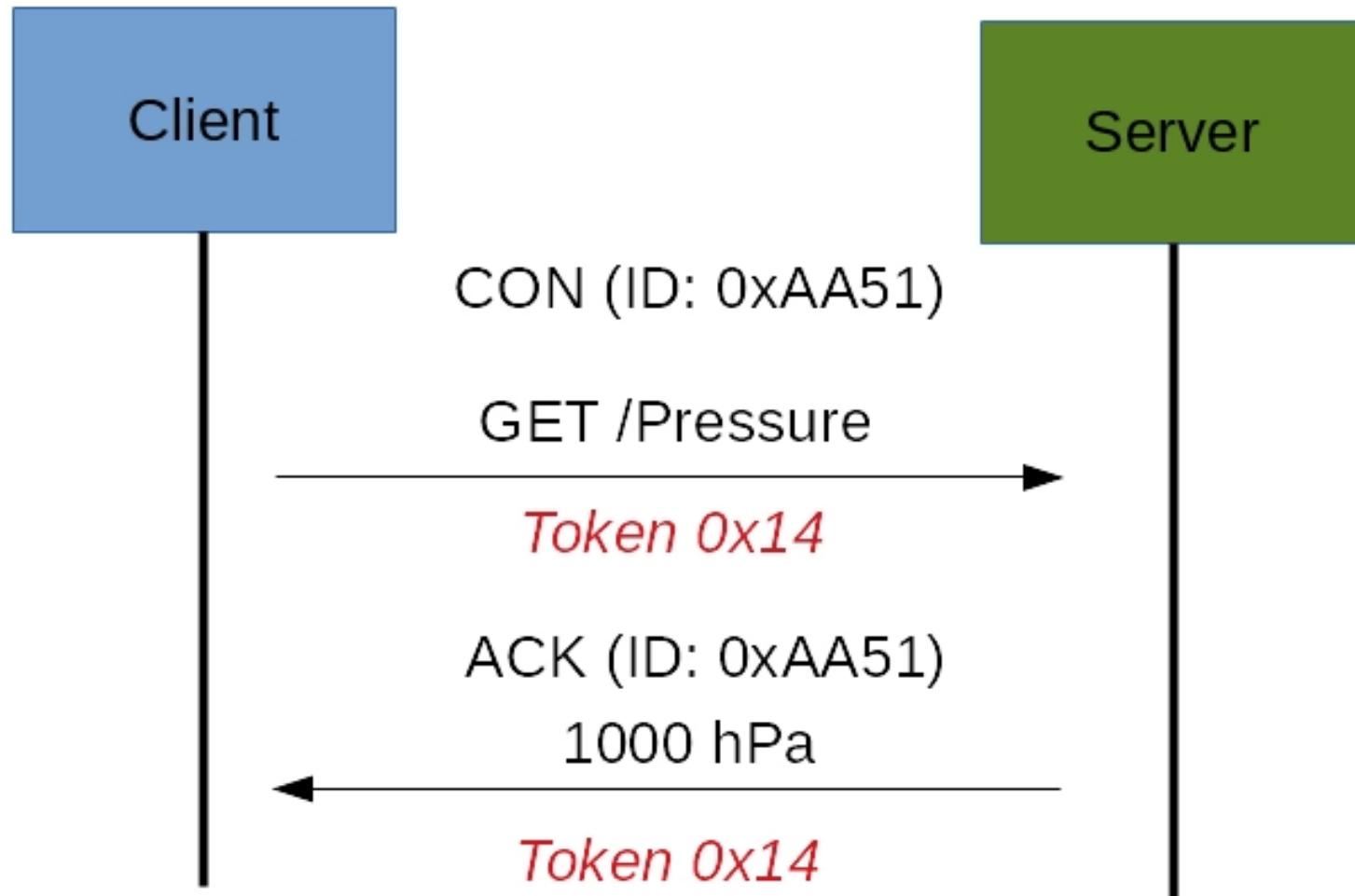
CoAp Request/Response Model

- The CoAP Request/Response is the second layer in the CoAP abstraction layer.
- The request is sent using a Confirmable (CON) or Non-Confirmable (NON) message.
- There are several scenarios depending on if the server can answer immediately to the client request or not.

CoAp Request/Response Model

- If the server can answer immediately to the client request, then if the request is carried using a Confirmable message (CON), the server sends back to the client an Acknowledge message containing the response or the error code.
- CoAP message use a Token. The Token is different from the Message-ID and it is used to match the request and the response.

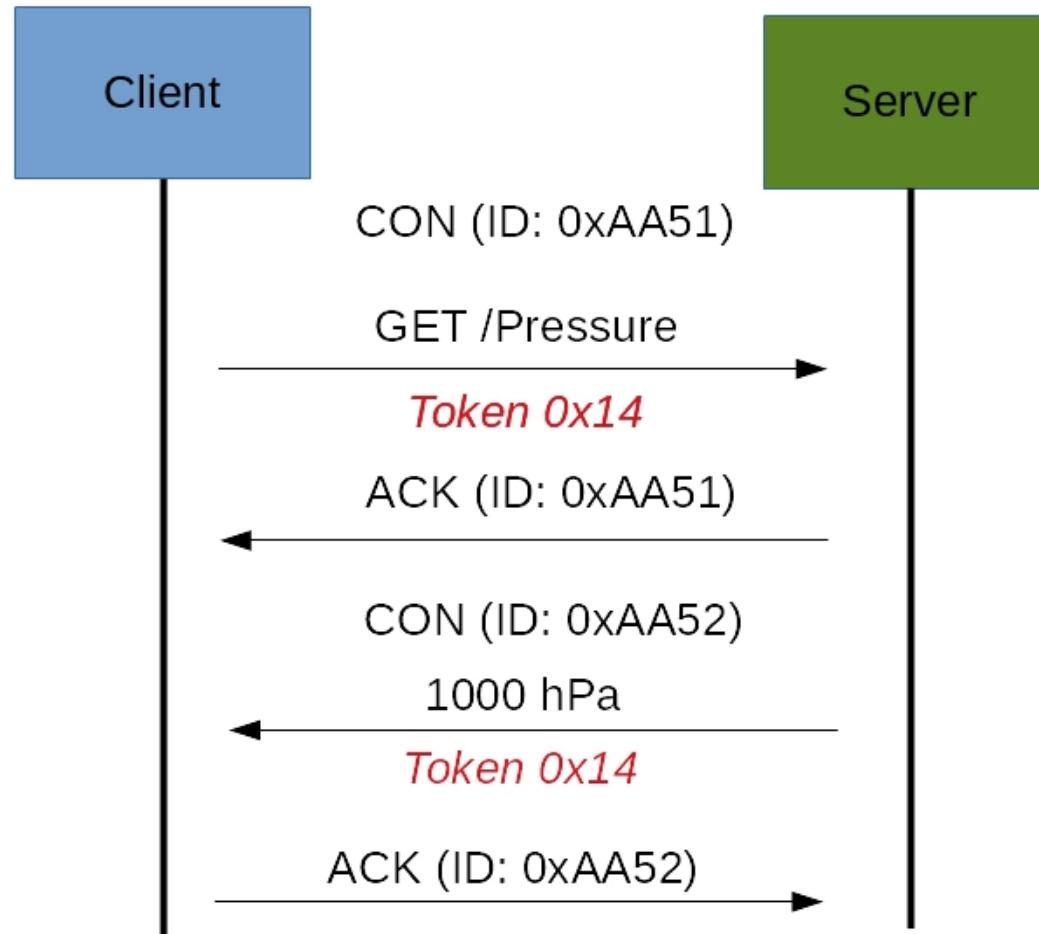
Example: CoAp Request/Response Model



CoAp Request/Response Model

- If the server can't answer to the request coming from the client immediately, then it sends an Acknowledge message with an empty response.
- As soon as the response is available, then the server sends a new Confirmable message to the client containing the response.
- At this point, the client sends back an Acknowledge message
- If the request coming from the client is carried using a NON-confirmable message, then the server answer using a NON-confirmable message.

Example: CoAp Request/Response Model



CoAP Message Format

- We have discussed different kinds of messages exchanged between the client and the server in CoAP protocol.
- The constrained application protocol is meant for constrained environments, and for this reason, it uses compact messages.
- To avoid fragmentation, a message occupies the data section of a UDP datagram. A message is made by several parts:

CoAP Message Format



CoAP Message Format

- **Ver:** It is a 2 bit unsigned integer indicating the version.
- **T:** it is a 2 bit unsigned integer indicating the message type: 0 confirmable, 1 non-confirmable
- **TKL:** Token Length is the token 4 bit length
- **Code:** It is the code response (8 bit length)
- **Message ID:** It is the message ID expressed with 16 bit.

CoAP Vs. MQTT

- MQTT is a protocol widely used in IoT.
- MQTT uses a publisher-subscriber while CoAP uses a request-response paradigm.
- MQTT uses a central broker to dispatch messages coming from the publisher to the clients.
- CoAP is essentially a one-to-one protocol very similar to the HTTP protocol.
- Moreover, MQTT is an event-oriented protocol while CoAP is more suitable for state transfer.
- COAP is simpler than HTTP, it will have lower latency and draw less power.

Internet of Things: DDS Protocol (Data Distribution Service), BLE

Lect #16

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

Data Distribution Service

- Data Distribution Service (DDS) is one of the messaging protocols being used in the IoT applications.
- It functions as a brokerless service.
- The approach used with DDS is the publish –subscribe model.
- DDS is a middleware protocol standard that supports enhanced data connectivity, improved reliability and boosted scalability for IIoT applications.
- Applications span from robotics, aerospace, military applications and domains where real-time data exchange is pivotal.

DDS Architecture and Functioning

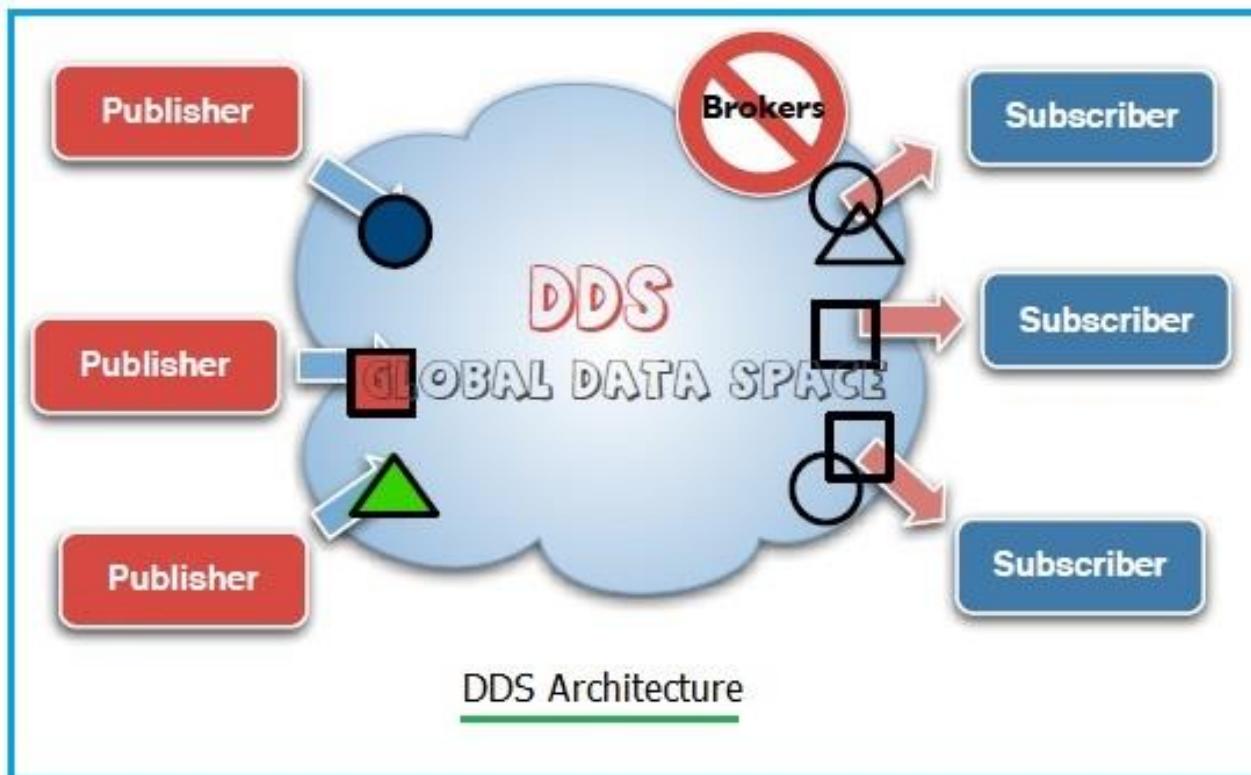
- As DDS is publish – subscribe model based architecture, event updates, sending and receiving data, commands within and among the nodes are all accomplished through the publish – subscribe approach.
- Similar to MQTT, TOPICs (pressure / temperature etc) are used.
- The publishing nodes should create the topics and publish the data.
- DDS should deliver the TOPICs to the subscribers.

DDS Architecture and Functioning

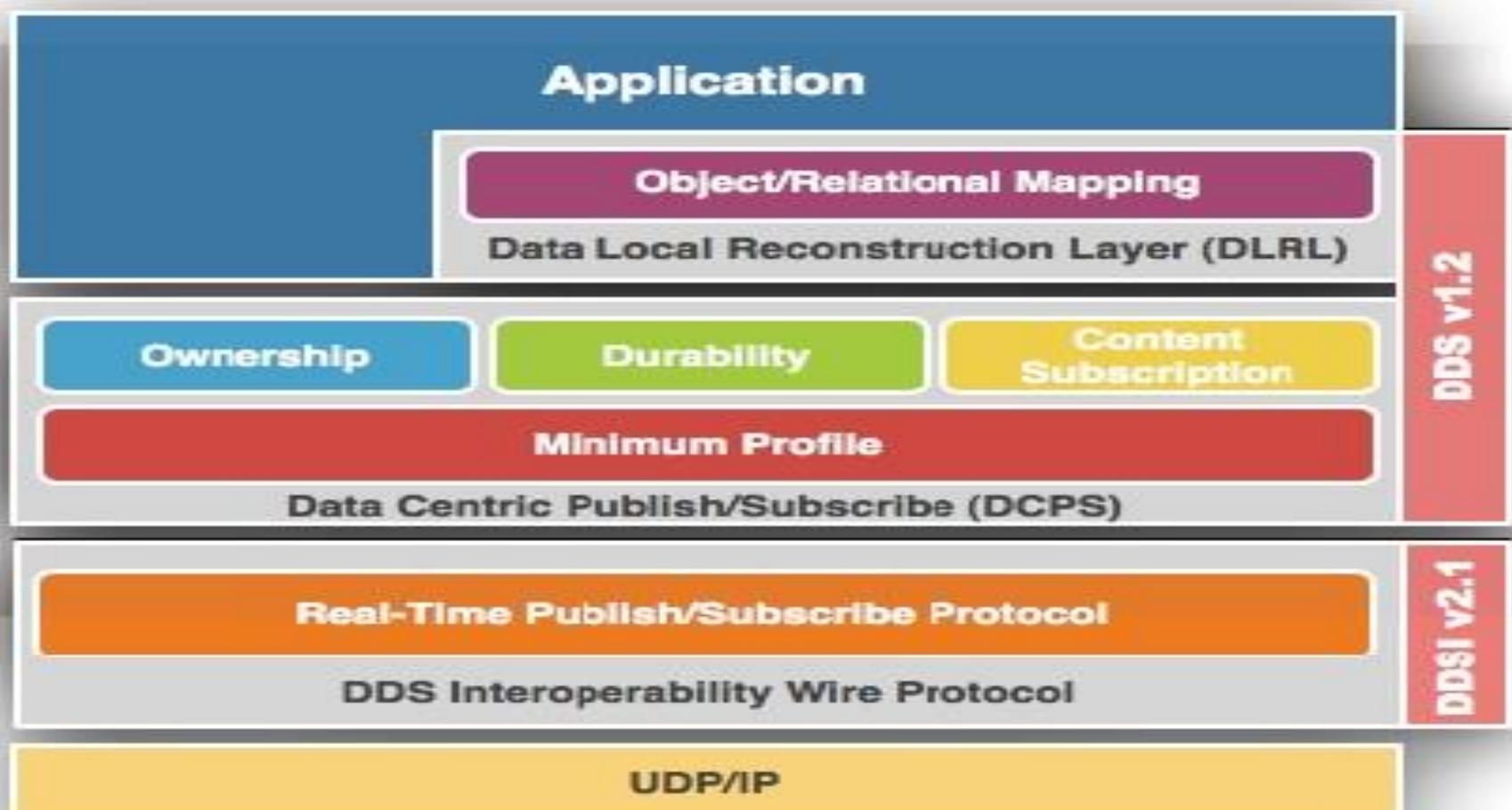
- DDS has to take care of all the attributes of data transfer. Addressing, marshalling and demarshalling data, flow control, and control of delivery are taken care by DDS.
- Marshalling refers to the process of converting the data or objects into a byte streaming, while unmarshalling is the process of converting the byte stream back to their original data or object.
- This enables subscribers on multiple platforms to receive the data from the publishers without hindrances.

DDS Architecture

- DDS is completely distributed global data space (GDS)



DDS Protocol Stack



DDS Protocol Stack

DDS Protocol Stack

- The data centric public-subscribe (DCPS) holds the responsibility for delivering data to the subscribers.
- The data local reconstruction layer (DLRL) provides Interface to the DCPS functionalities.
- DLRL helps in sharing the data within the IoT devices.
- DDS V1.2 is an API that is language independent, and hardware and OS independent . This also ensures the code portability across various available platforms
- DDSI V2.1 ensures the wire interoperability between different available implementations of the DDS.

Transport Protocol: Bluetooth Low Energy (BLE)

- BLE is an open low-energy short-range radio communication technology.
- It is also called Bluetooth Smart.
- The protocol is designed by Bluetooth Special Interest Group (SIG).
- It is a wireless personal area network (PAN) technology same as Bluetooth.
- Symbol:



BLE over Bluetooth

- BLE has reduced power consumption, as compared to classic Bluetooth. (power saving)
- BLE applications span from healthcare, entertainment, fitness, proximity related applications (car lock etc).
- BLE is supported by Android, IOS, Blackberry etc making it easier to use with any mobile OS.
- BLE is supported by macOS, windows 8 / 10 and linux variants.

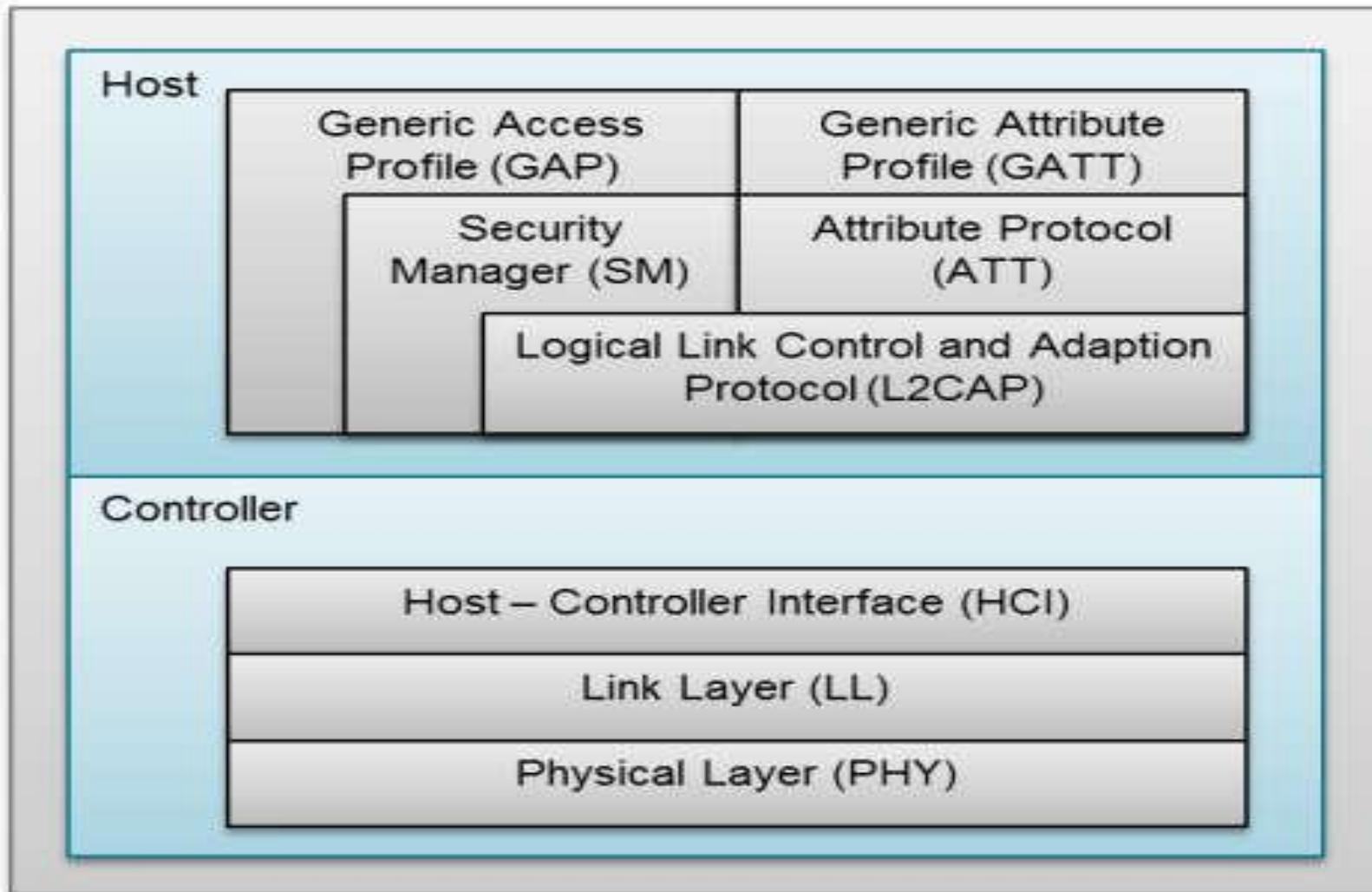
BLE Features

- It is license free and hence does not add any costing related overhead to the system.
- There is no restrictions with respect to manufacturers. Any system manufactured by any manufacturer is fine with BLE.
- These are very inexpensive and much affordable.
- The size of BLE chips are very small and hence does not occupy much space on board.
- Power consumption is very minimal and hence is very suitable for IoT applications
- The range offered by BLE is much higher and better than the previous versions.

BLE components

- BLE has three components from its architectural perspective:
 - Application block (user application is situated here. It interacts directly with bluetooth stack)
 - Host block (upper layer of bluetooth stack)
 - Controller block (lower layer of bluetooth stack)
 - Host Controller Interface (interface the controller with the host)

BLE Protocol stack



Internet of Things: BLE (Bluetooth Low Energy)

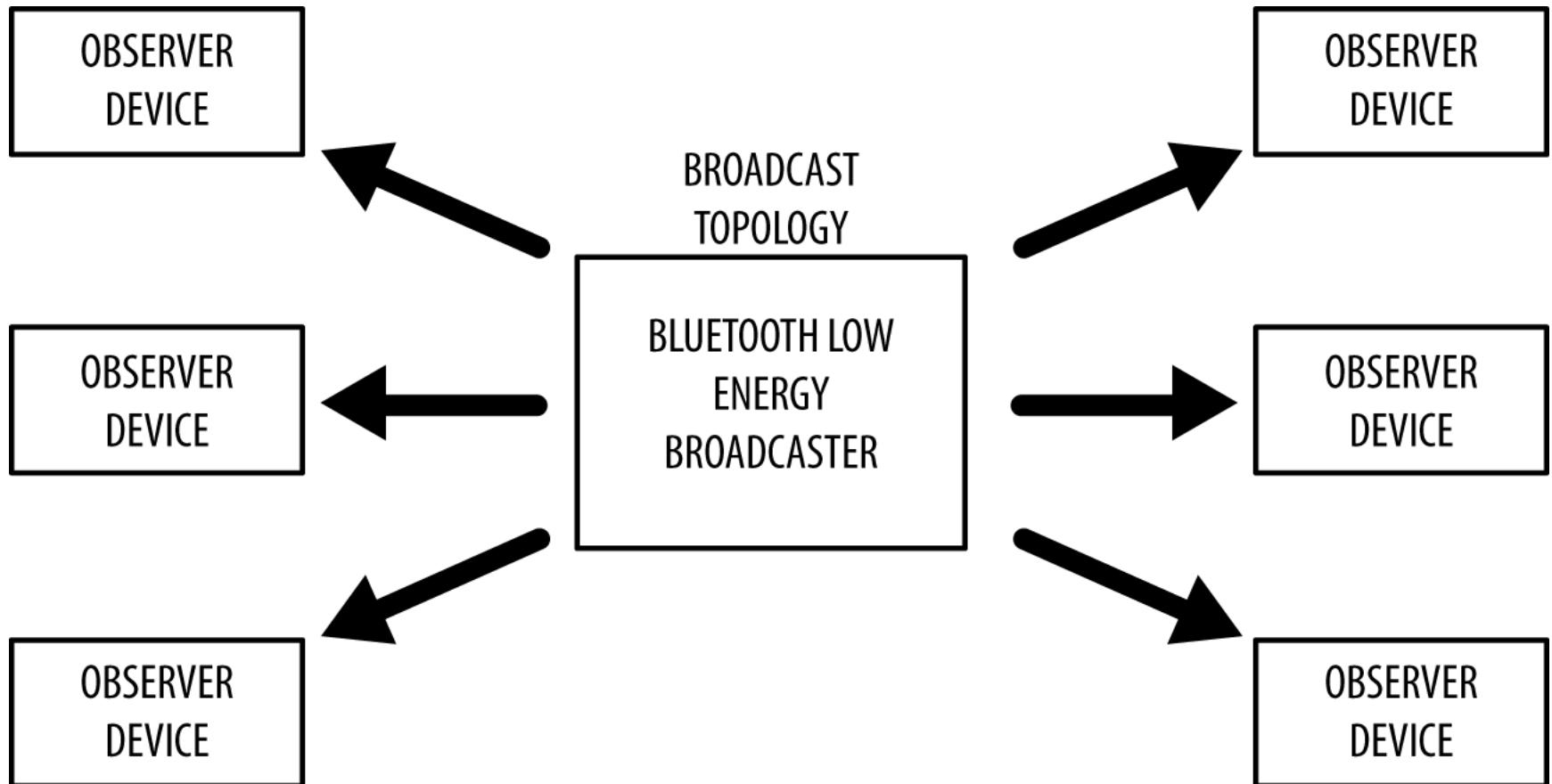
Lect #17

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

BLE Topology

- A BLE device communicates with outside world by broadcasting and connections.
- Broadcasting:
 - With broadcast, the data is sent out “One way”.
 - Whichever device is capable of receiving the data, will receive it.
 - This is analogous to a radio broadcast, where you pickup the content aired by tuning the channels.
 - The broadcaster sends out the “non-connectable” advertising packets in frequent intervals to all those who are willing to collect it.

Bluetooth broadcasting

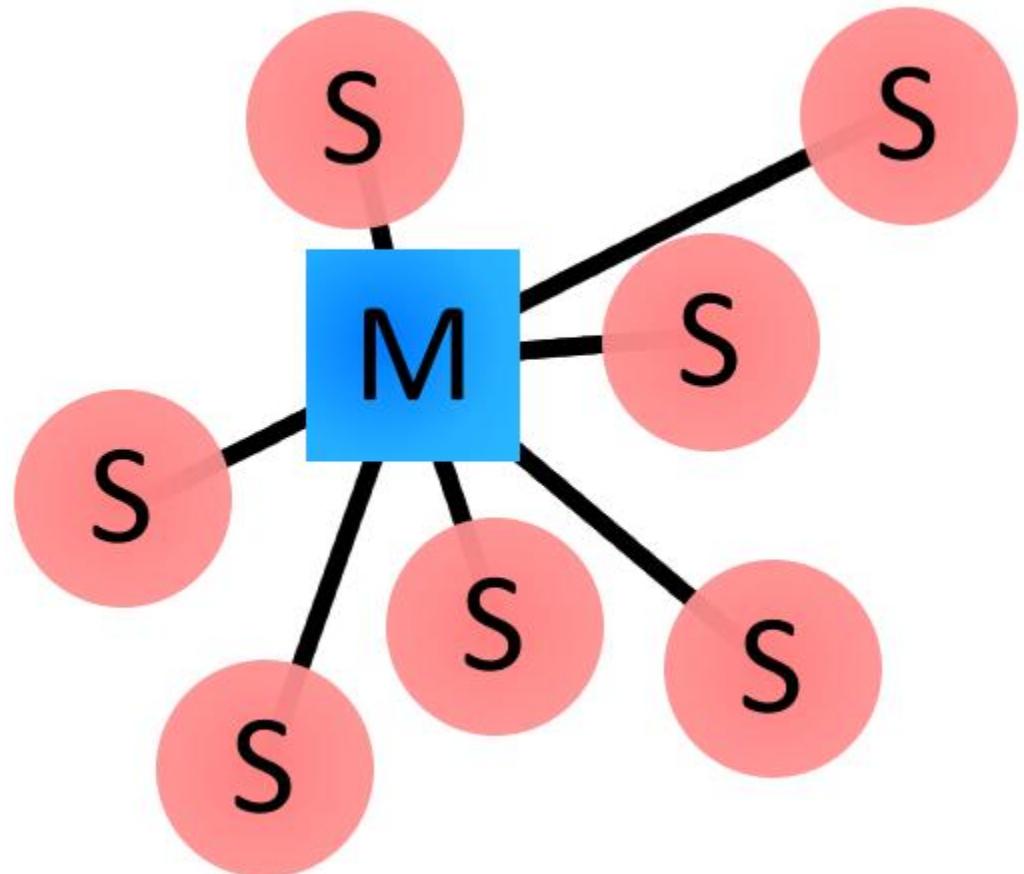
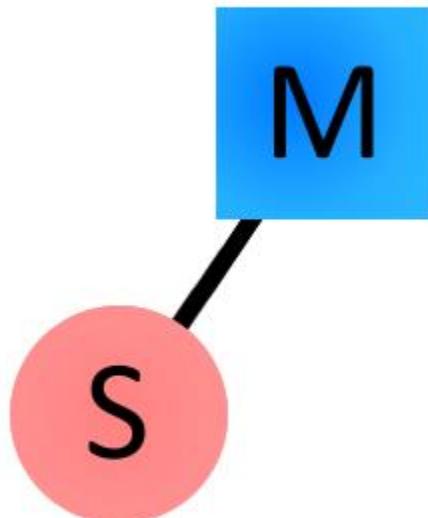


- On the other hand, the observer will keep scanning at the pre-set frequency to receive any non-connectable packets that are being broadcasted.

Connections

- This is the second possible way of communication.
- A connection is permanent and enables periodical data transfer between two devices.
- The term master-slave configuration comes into picture in this case.
- The master will try to find out the connectable advertising packets.
- If found, the connection will be initiated.

Connections



Connections

- After the connection is established, the master will take care of the periodic data exchange.
- The slave will keep periodically sending the connectable advertising packets.
- When there is an incoming request, it would be accepted by the slave to whom it is addressed.

Connections

- When two devices are connected, it is called connection event.
- This is ideal for power saving.
- If required, the device will work and enable high speed process.
- Power saving is facilitated by not invoking the device when not required.
- To make it precise, the scheme only works when there is a need.

Connections

- The rest of the time, the devices are in sleep mode.
- However, communication still happens when the device is in sleep mode as well, thereby saving power.
- In sleep mode, the device is invoked only when data has to be transmitted.

Classic bluetooth versus BLE

Specifications	Classic Bluetooth	Bluetooth Low Energy (BLE)
Range	100 m	Greater than 100 m
Data Rate	1-3 Mbps	1 Mbps
Application Throughput	0.7 -2.1 Mbps	0.27 Mbps
Frequency	2.4 GHz	2.4 GHz
Security	56/128-bit	128-bit AES with Counter Mode CBC-MAC
Robustness	Adaptive fast frequency hopping, FEC, fast ASK	24-bit CRC, 32-bit Message Integrity Check
Latency	100 ms	6 ms
Time Lag	100 ms	3 ms
Voice Capable	Yes	No
Network Topology	Star	Star
Power Consumption	1 W	0.01 to 0.5 W
Peak Current Consumption	less than 30 mA	less than 15 mA

IoT Protocol Categories

Infrastructure protocols	Application protocols		DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP REST			
	Service discovery		mDNS				DNS-SD					
	Routing protocol	RPL										
	Network layer	6LoWPAN				IPv4/IPv6						
	Link layer	IEEE 802.15.4										
	Physical/device layer	LTE-A	EPCglobal	IEEE 802.15.4	Z-Wave							

Routing Protocol: RPL

- RPL stands for routing protocol for low power and lossy networks.
- It is an IPv6 protocol.
- Low power lossy networks include wireless personal area networks (WPANs), low-power line communication (PLC) networks, and wireless sensor networks (WSNs).

Routing Protocol: RPL

- These networks have some characteristics:
 - Capability to optimize and save energy
 - Capability to support traffic patterns other than unicast communication
 - Capability to run routing protocols over link layers with restricted frame sizes

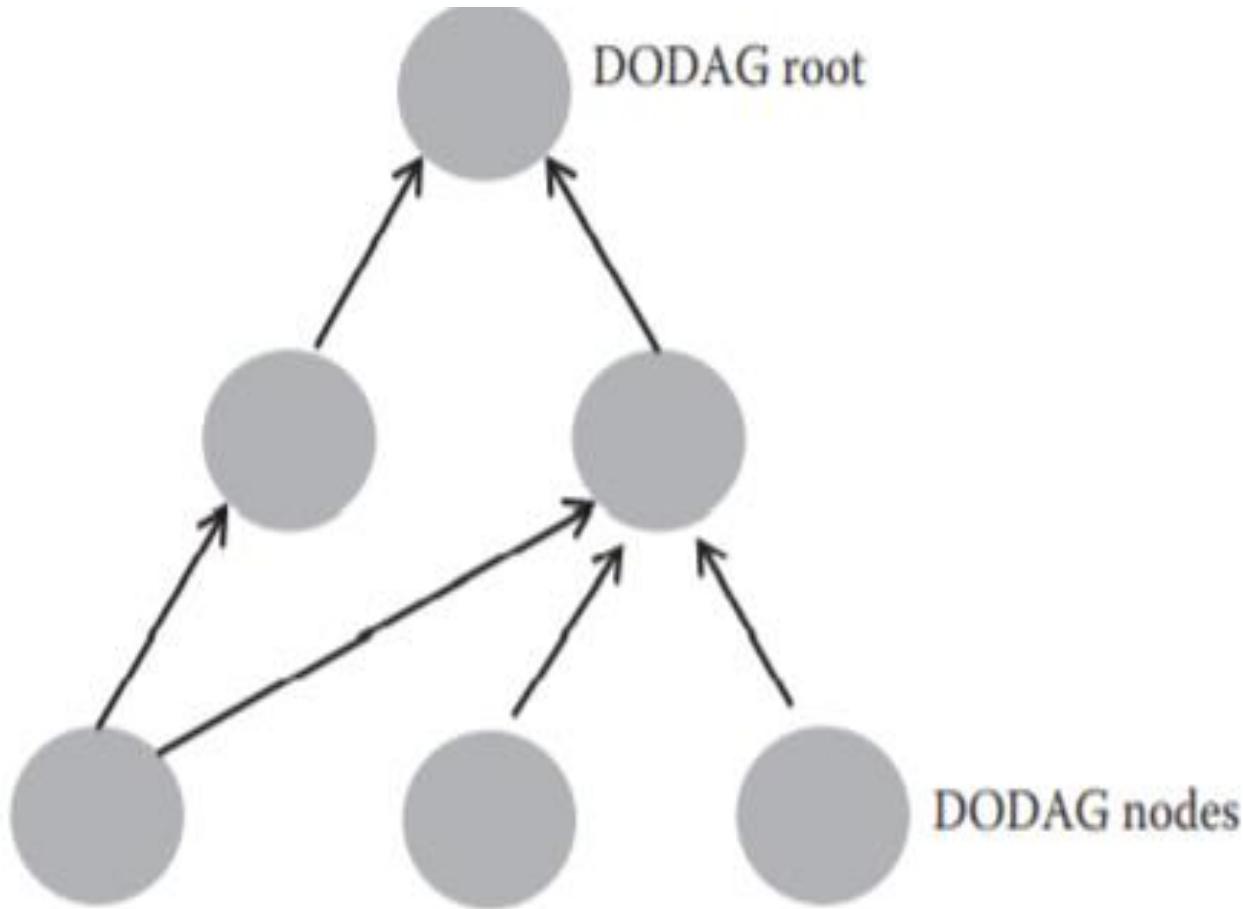
Routing Protocol: RPL

- RPL was designed to support minimal routing needs by building a highly robust topology over lossy networks.
- This protocol provides support for various types of traffic models: multipoint-to-point, point-to-multipoint, and point-to-point.
- Devices in the network that use this protocol are connected to each other in such a way that no cycles are present in the connection.

Routing Protocol: RPL

- In order to achieve this, a destination oriented directed acyclic graph (DODAG), which is routed at a single destination, is built initially.
- RPL specifications refer to DODAG as DODAG root.
- Each node that is a part of DODAG knows its parent node but does not have any information about its child nodes.
- RPL maintains at least a single path from each node to the root and the preferred parent.
- This is done in order to increase performance by pursuing a faster path.

DODAG topology



DODAG control messages

- The four important types of control messages are used by RPL to maintain routing topology and maintain the updated routing information.
- The control messages are:
 - **DODAG information object (DIO)**
 - **Destination advertisement object (DAO)**
 - **DODAG information solicitation (DIS)**
 - **DAO acknowledgment (DAO-ACK)**

DODAG information object : DIO

- Used to keep the current rank /level of the node
- Determine the distance of each node to the root
- Choose the preferred parent path
- The DODAG Information Object (DIO) carries information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG

DAO

- Used to unicast destination information toward selected parents of a node.
- Helps RPL to maintain upward and downward traffic.
- The Destination Advertisement Object (DAO) message is used to propagate reverse route information to record the nodes visited along the upward path.
- A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root
- Rank strictly increases in the down direction and strictly decreases in the up direction.

DAO

- The exact way Rank is computed depending on the DAG's Objective Function (OF).
- The Rank may analogously track a simple topological distance, may be calculated as a function of link metrics, and may consider other properties such as constraints.

DIS

- **Used by a specific node in order to acquire DIO messages from another reachable adjacent node.**

DAO-ACK

- **Used as a response to a DAO msg and is sent by a DAO recipient node like a DAO parent or DODAG root.**

<i>Serial Number</i>	<i>Name of the Message</i>	<i>Description</i>
1	DODAG information object (DIO)	This message is used to keep the current rank (level) of the node, determine the distance of each node to the root based on some specific metrics, and choose the preferred parent path.
2	Destination advertisement object (DAO)	This message is used to unicast destination information toward selected parents of a node. This control message helps RPL to maintain upward and downward traffic.
3	DODAG information solicitation (DIS)	This message is used by a specific node in order to acquire DIO messages from another reachable adjacent node.
4	DAO acknowledgment (DAO-ACK)	This message is used as a response to a DAO message and is sent by a DAO recipient node like a DAO parent or DODAG root.

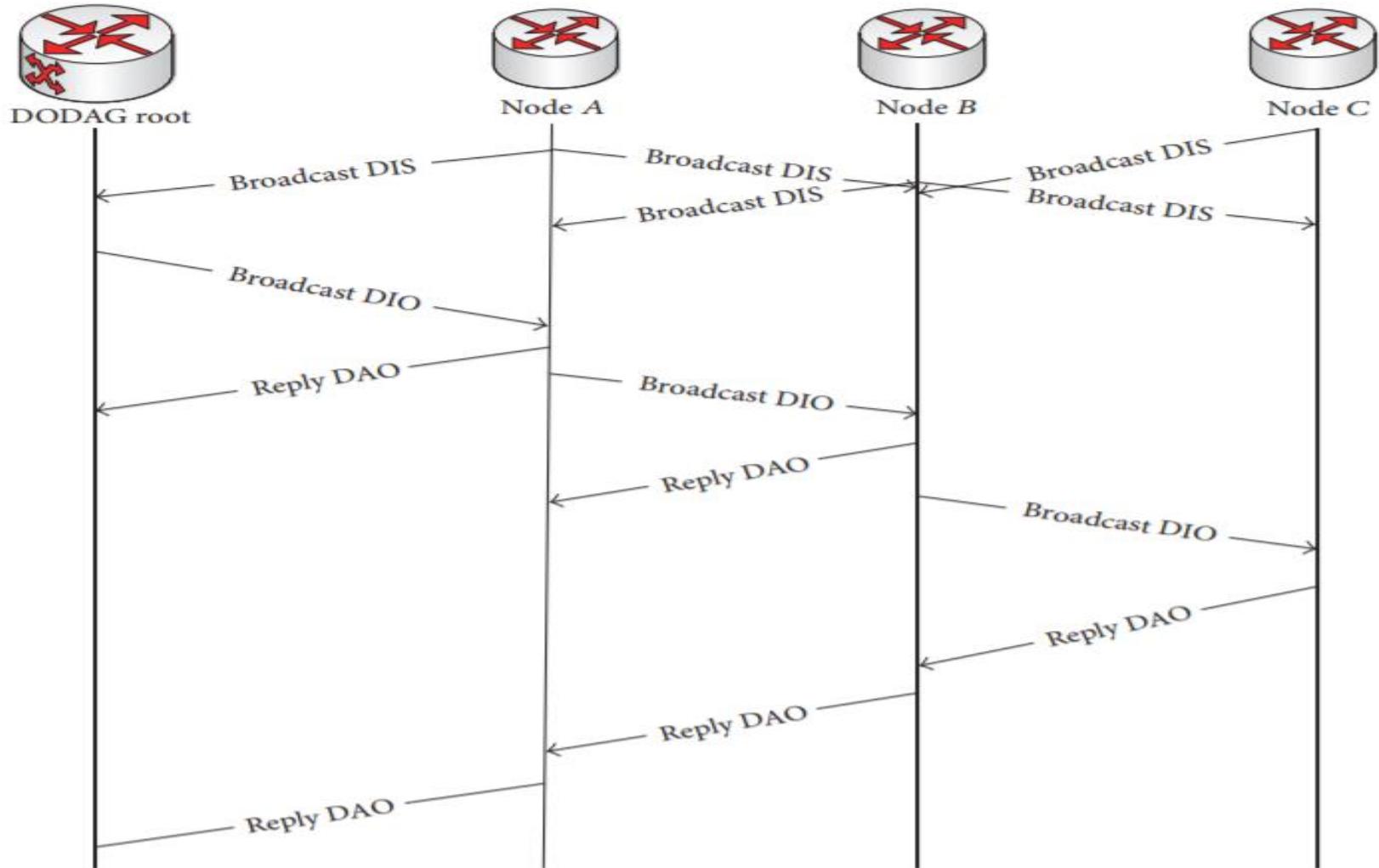
DODAG Creation

- A DODAG will start its formation when the root node starts sending its location information using DIO message to all low-power lossy network (LLN) levels.
- The process of the DODAG construction begins from the DODAG root.
- First, the DODAG root broadcasts the DODAG information by transmitting a DIO message.
- Neighbors of the root receive and process the DIO message.
- The DIO message is processed and transmitted to other nodes one by one.

DODAG creation

- A node which has not joined any DODAGs calculates the cost of the path to the DIO sender and then decides whether to join the DODAG or not.
- Once a neighbor joins the DODAG, it has a route towards the DODAG root, and the root becomes a DODAG parent of the node.
- Next, the node calculates its Rank in the DODAG and replies with a DAO message to its parent to inform its participation.
- A node which has not received any DIO messages and has not joined any DODAGs can request DODAG information by sending DIS messages periodically to its neighbors.
- All of the neighbors repeat this process until all of the nodes join the DODAG.

Flow diagram of DODAG construction



- The DODAG root sends a DIO message with DODAG information;
- node A receives the DIO message and joins the DODAG and replies with a DAO message which has its prefix information to the root.
- After that node A sends a DIO message which contains the DODAG information and node B which is in node A's transmission scope receives the message, joins the DODAG, and replies with a DAO message to node A.
- Node B has received a DIS request message from node C but replies nothing because it has not joined any DODAG.

- After node B joined the DODAG, node B sends DIO message to node C to invite it to join the DODAG.
- Node C replies with a DAO message to node B after it has joined the DODAG.
- Node B will integrate information after receiving the message and then send the DAO message to its preferred parent node A.
- After all, the DODAG root acquires all nodes' prefix information by their DAO messages to form a downward route.

Internet of Things: Distance Vector, DSDV Protocol

Lect #18

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

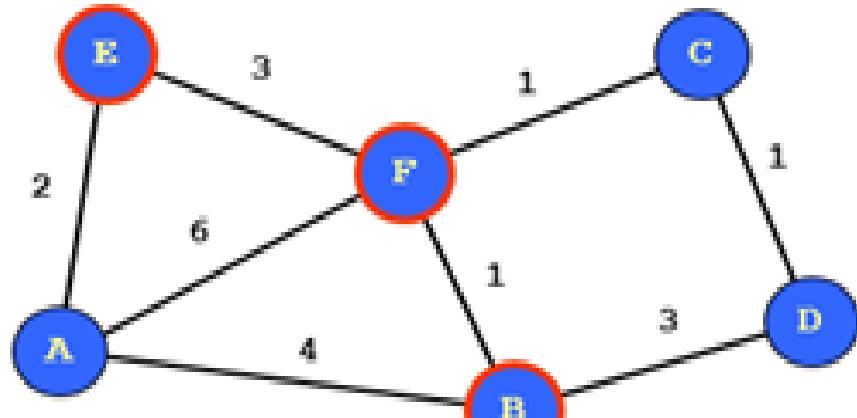
Distance-Vector

- known also as Distributed Bellman-Ford or RIP (Routing Information Protocol)
- Every node maintains a routing table
 - all available destinations
 - the next node to reach to destination
 - the number of hops to reach the destination
- Periodically send table to all neighbors to maintain topology

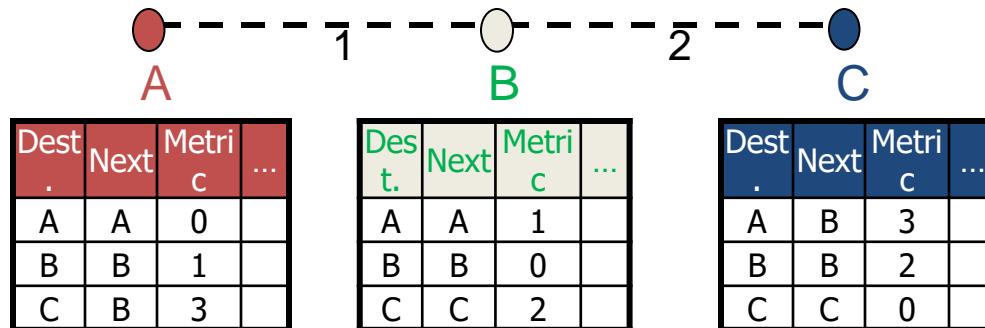
Example: Distance Vector Algorithm

Table for A			Table for B		
Dst	Cst	Hop	Dst	Cst	Hop
A	0	A	A	4	A
B	4	B	B	0	B
C	6	E	C	2	F
D	7	B	D	3	D
E	2	E	E	4	F
F	5	E	F	1	F

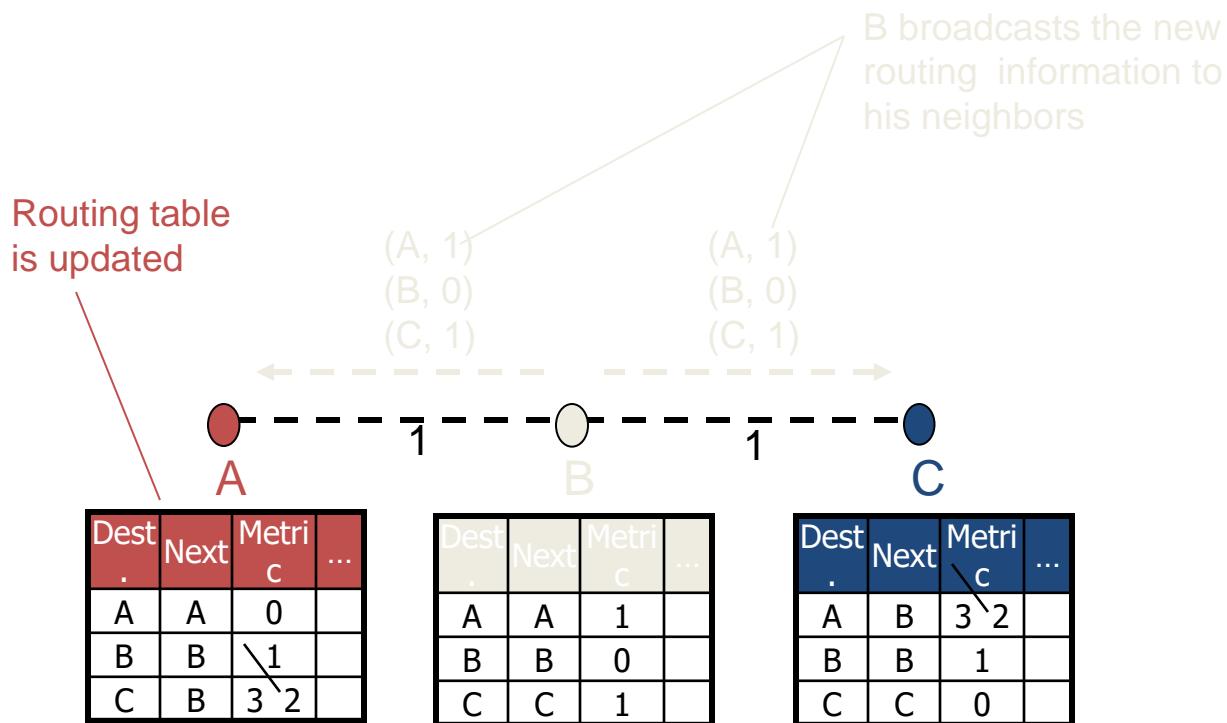
Table for C			Table for D			Table for E			Table for F		
Dst	Cst	Hop									
A	6	F	A	7	B	A	2	A	A	5	B
B	2	F	B	3	B	B	4	F	B	1	B
C	0	C	C	1	C	C	4	F	C	1	C
D	1	D	D	0	D	D	5	F	D	2	C
E	4	F	E	5	C	E	0	E	E	3	E
F	1	F	F	2	C	F	3	F	F	0	F



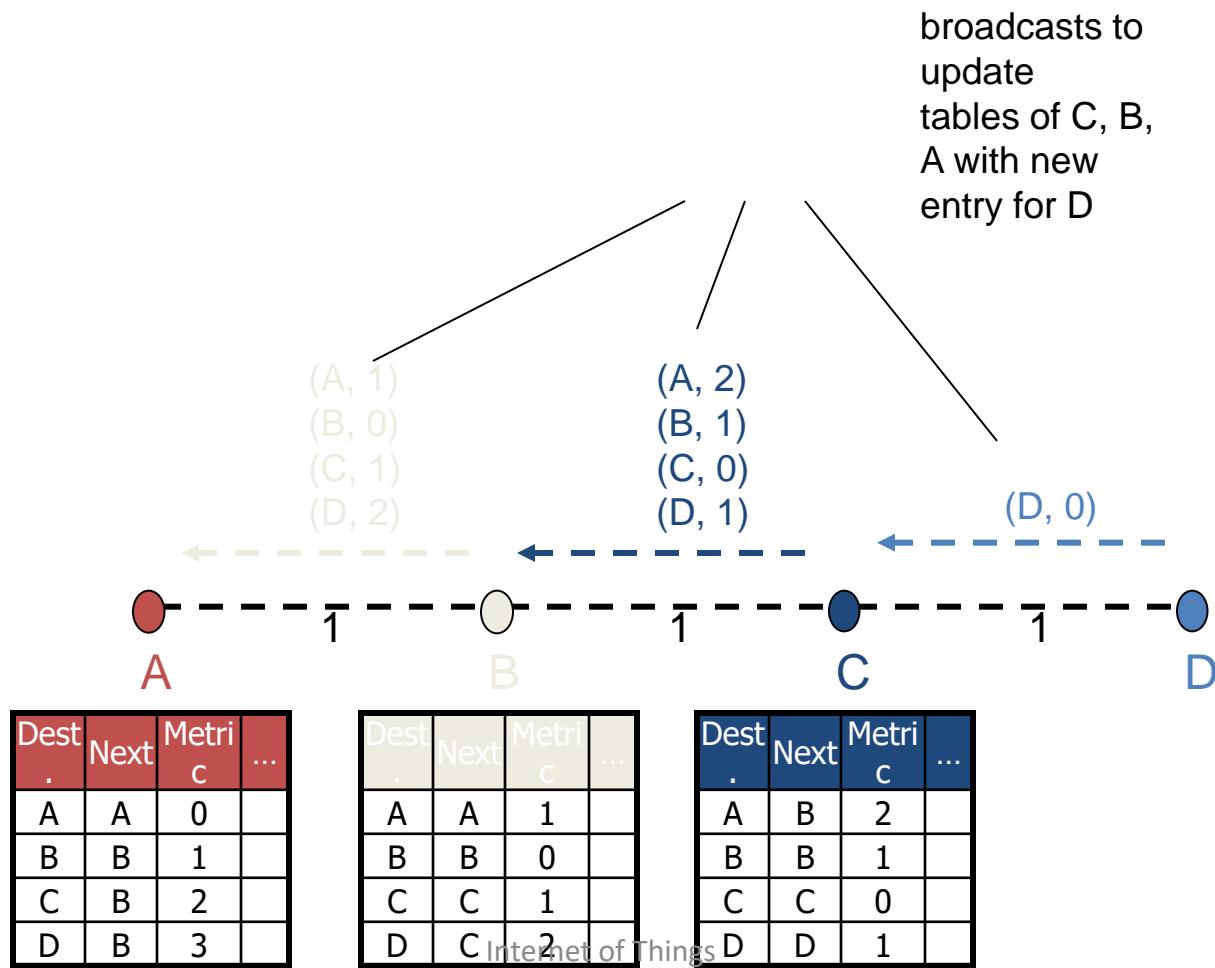
Distance Vector (Tables)



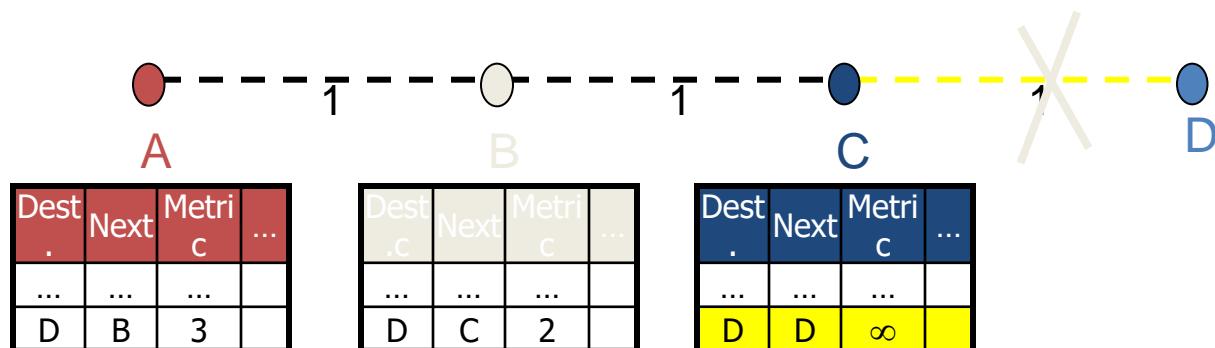
Distance Vector (Update)



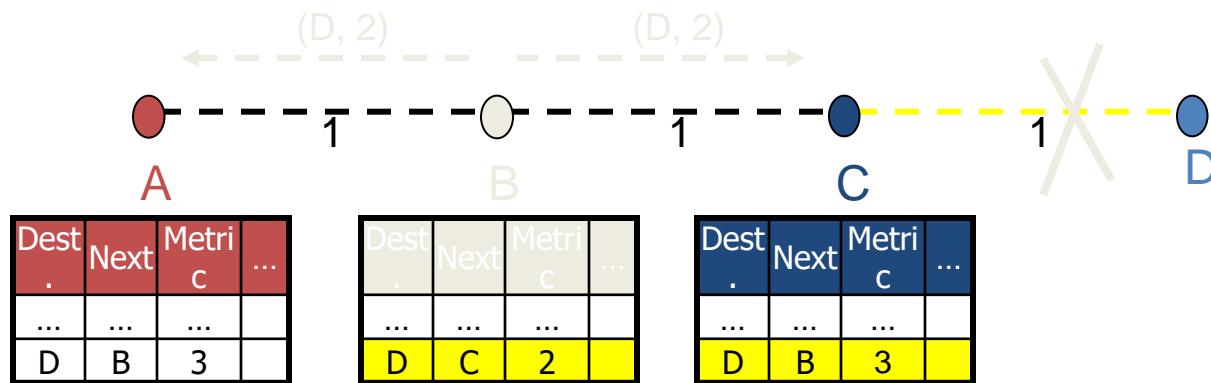
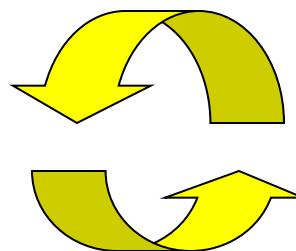
Distance Vector (New Node)



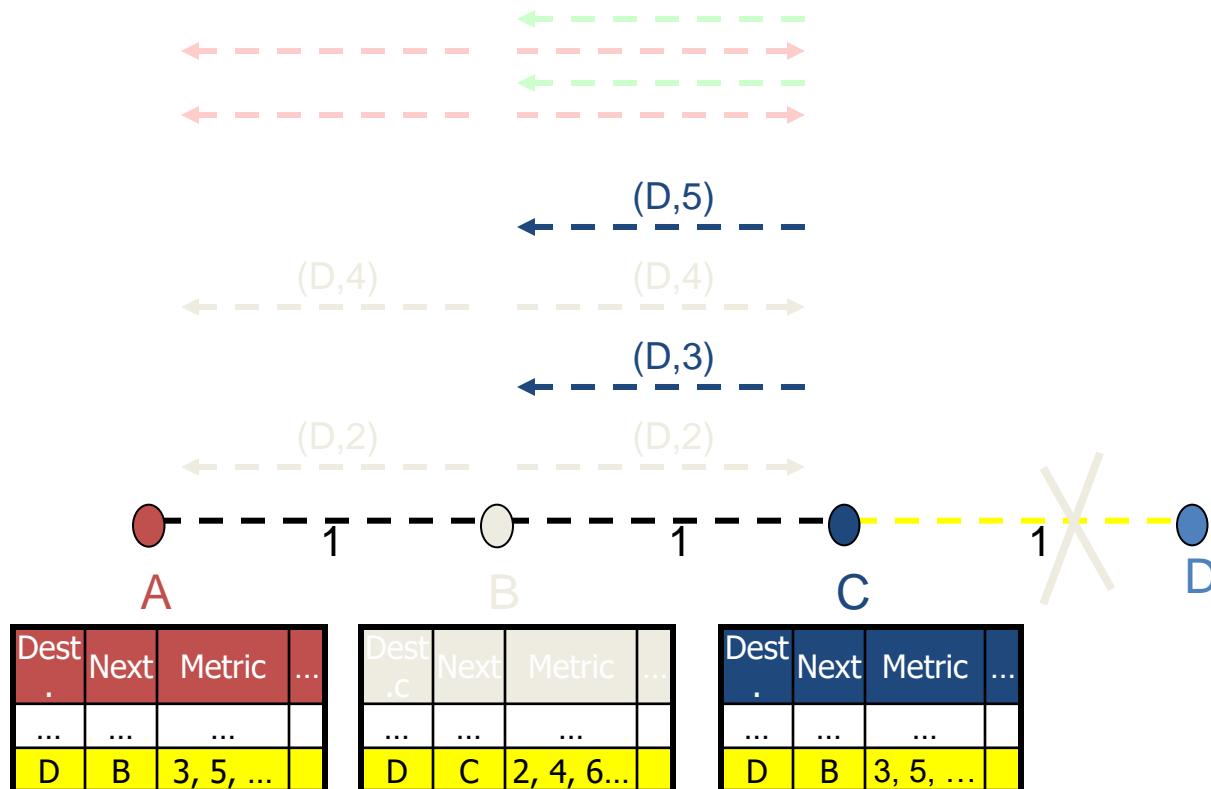
Distance Vector (Broken Link)



Distance Vector (Loops)



Distance Vector (Count to Infinity)



Distance Vector

- DV not suited for ad-hoc networks!
 - Loops
 - Count to Infinity
- New Solution -> DSDV Protocol

DSDV Protocol

- DSDV is Destination Based
- No global view of topology

DSDV Protocol

- DSDV is Proactive (Table Driven)
 - Each node maintains routing information for all known destinations
 - Routing information must be updated periodically
 - Traffic overhead even if there is no change in network topology
 - Maintains routes which are never used

DSDV Protocol

- Keep the simplicity of Distance Vector
- Guarantee Loop Freeness
 - New Table Entry for Destination Sequence Number
- Allow fast reaction to topology changes
 - Make immediate route advertisement on significant changes in routing table
 - but wait with advertising of unstable routes (damping fluctuations)

DSDV: Step 1- Table Entries

- **Sequence number** originated from destination. Ensures loop freeness.
- **Install Time** when entry was made (used to delete stale entries from table)
- **Stable Data** Pointer to a table holding information on how stable a route is. Used to damp fluctuations in network.

Destination	Next	Metric	Seq. Nr	Install Time	Stable Data
A	A	0	A-550	001000	Ptr_A
B	B	1	B-102	001200	Ptr_B
C	B	3	C-588	001200	Ptr_C
D	B	4	D-312	001200	Ptr_D

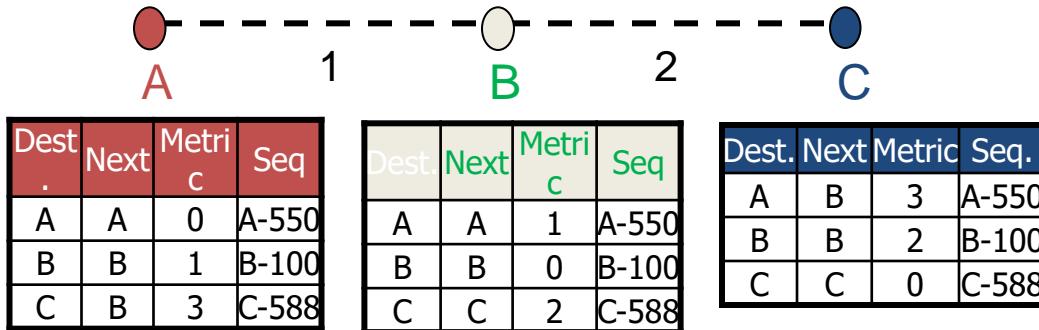
DSDV: Step 2- Route Advertisements

- Advertise to each neighbor own routing information
 - Destination Address
 - Metric = Number of Hops to Destination
 - Destination Sequence Number
- Rules to set sequence number information
 - On each advertisement increase own destination sequence number (use only even numbers)
 - If a node is no more reachable (timeout) increase sequence number of this node by 1 (odd sequence number) and set metric = ∞

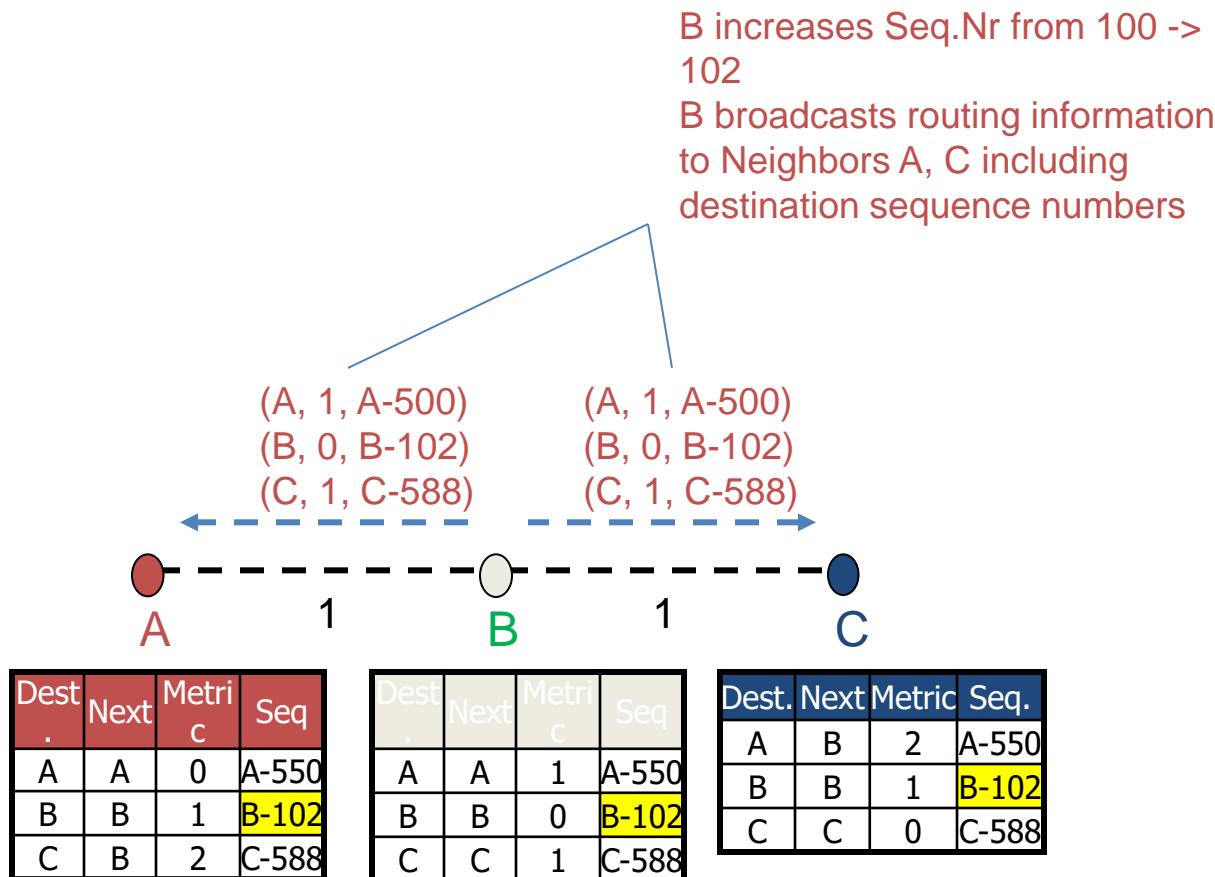
DSDV: Step 3 - Route Selection

- Update information is compared to own routing table
 - 1. Select route with higher destination sequence number (This ensure to use always newest information from destination)
 - 2. Select the route with better metric when sequence numbers are equal.

DSDV (Tables)



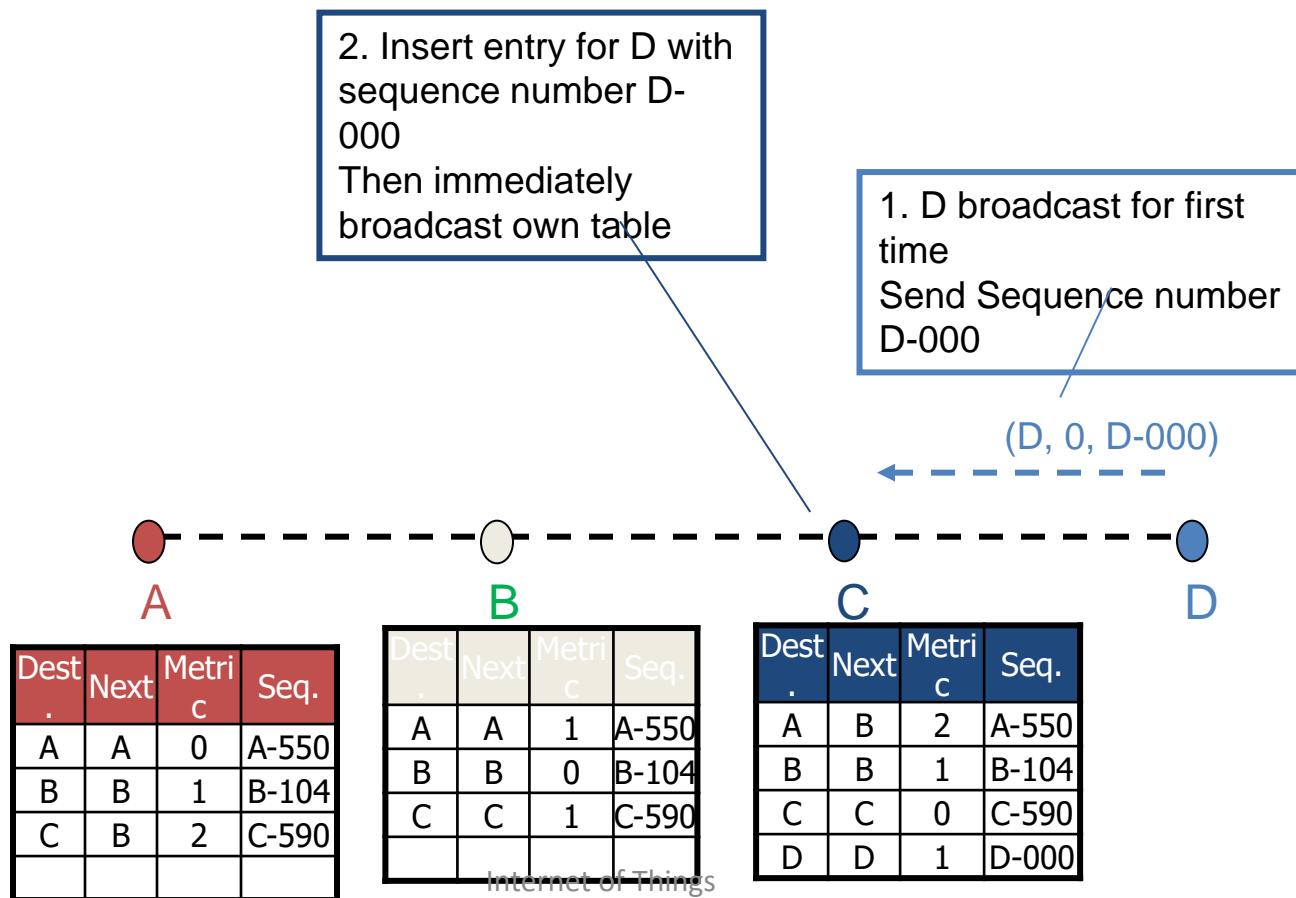
DSDV (Route Advertisement)



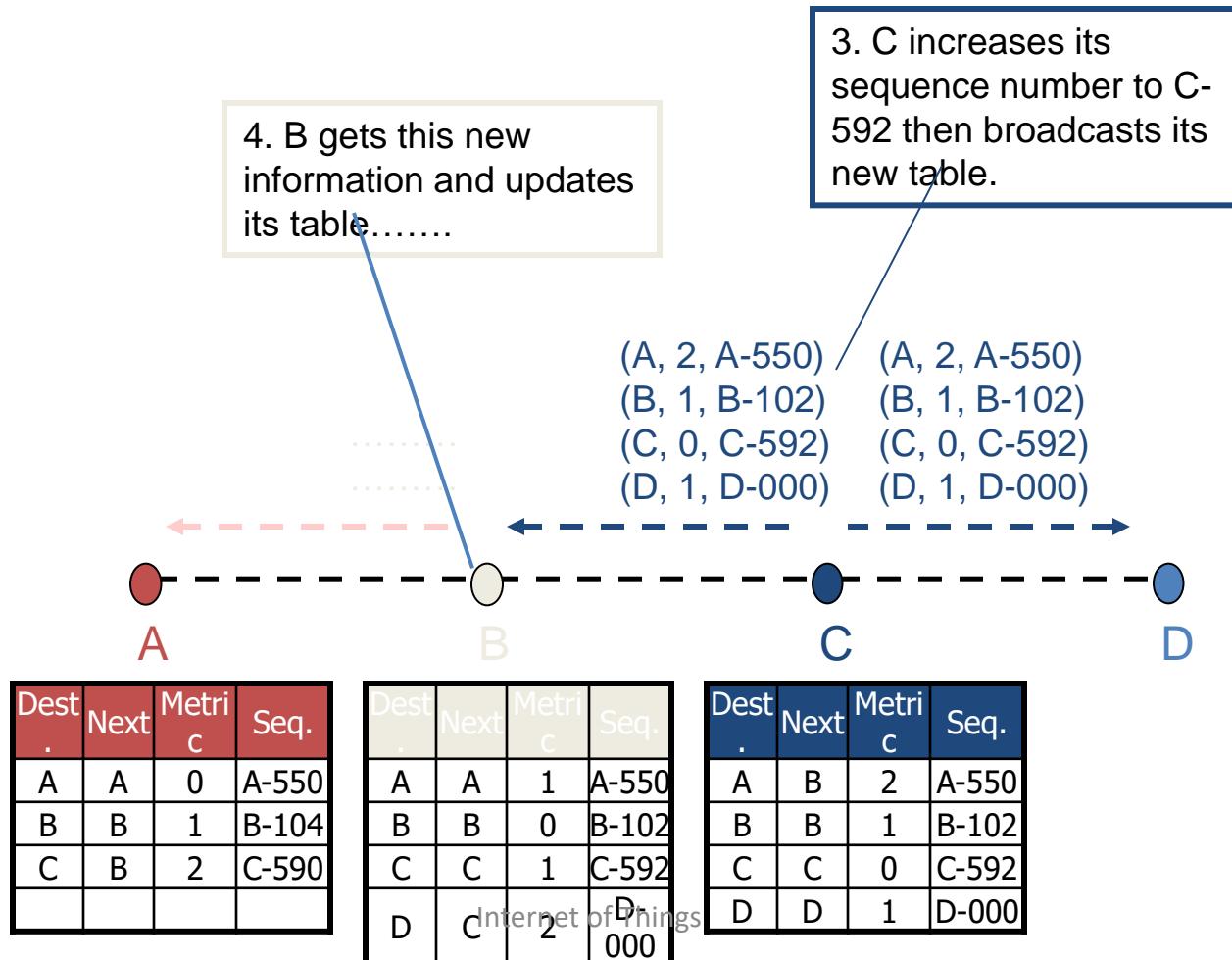
DSDV :Respond to Topology Changes

- Immediate advertisements
 - Information on new Routes, broken Links, metric change is immediately propagated to neighbors.
- Full/Incremental Update:
 - Full Update: Send all routing information from own table.
 - Incremental Update: Send only entries that has changed. (Make it fit into one single packet)

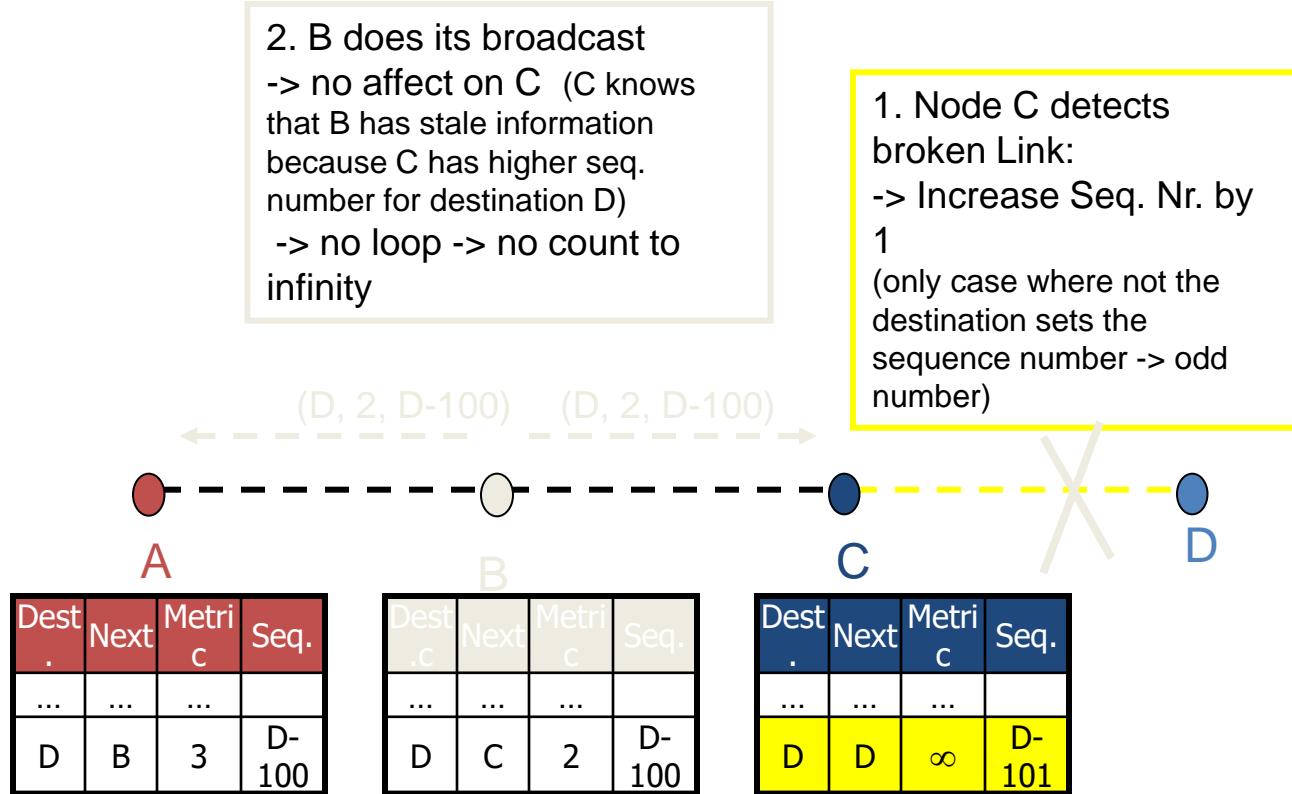
DSDV: New Node Entry



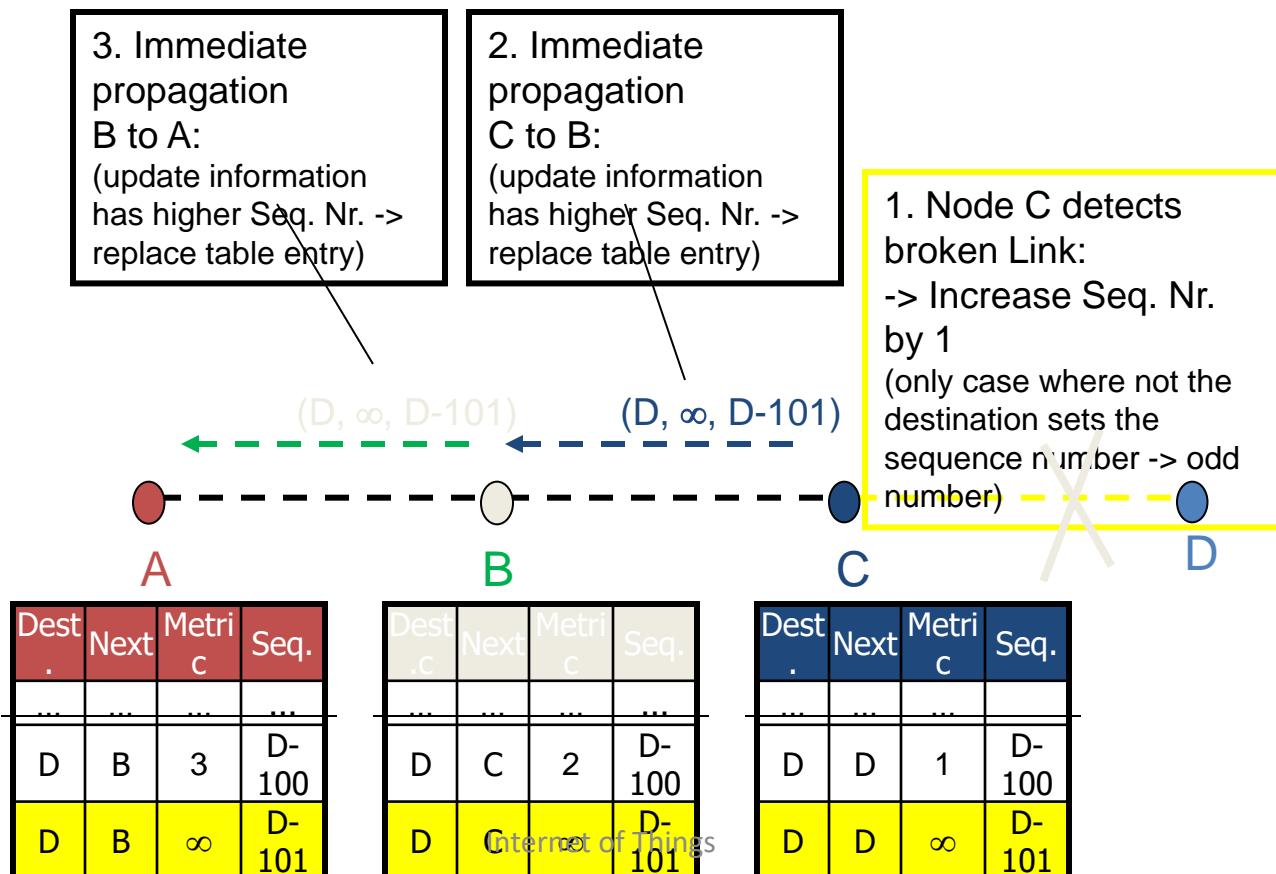
DSDV: New Node Entry



DSDV :no loops, no count to infinity



DSDV: Immediate Advertisement



Summery

- Advantages
 - Simple (almost like Distance Vector)
 - Loop free through destination seq. numbers
 - No latency caused by route discovery
- Disadvantages
 - No sleeping nodes
 - Overhead: most routing information never used

Internet of Things: RPL Protocol

Lect #19

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

Low power and lossy network (LLN)

- LLN typically composed of many embedded devices with limited power, memory, and processing resources. (constrained nodes)
- The devices are interconnected by a variety of links, such as IEEE 802.15.4 (LR-WPAN) or low-power Wi-Fi.
- There is a wide scope of application areas for LLNs, including industrial monitoring, building automation, connected home, health care, environmental monitoring, urban sensor networks, energy management, assets tracking, etc.

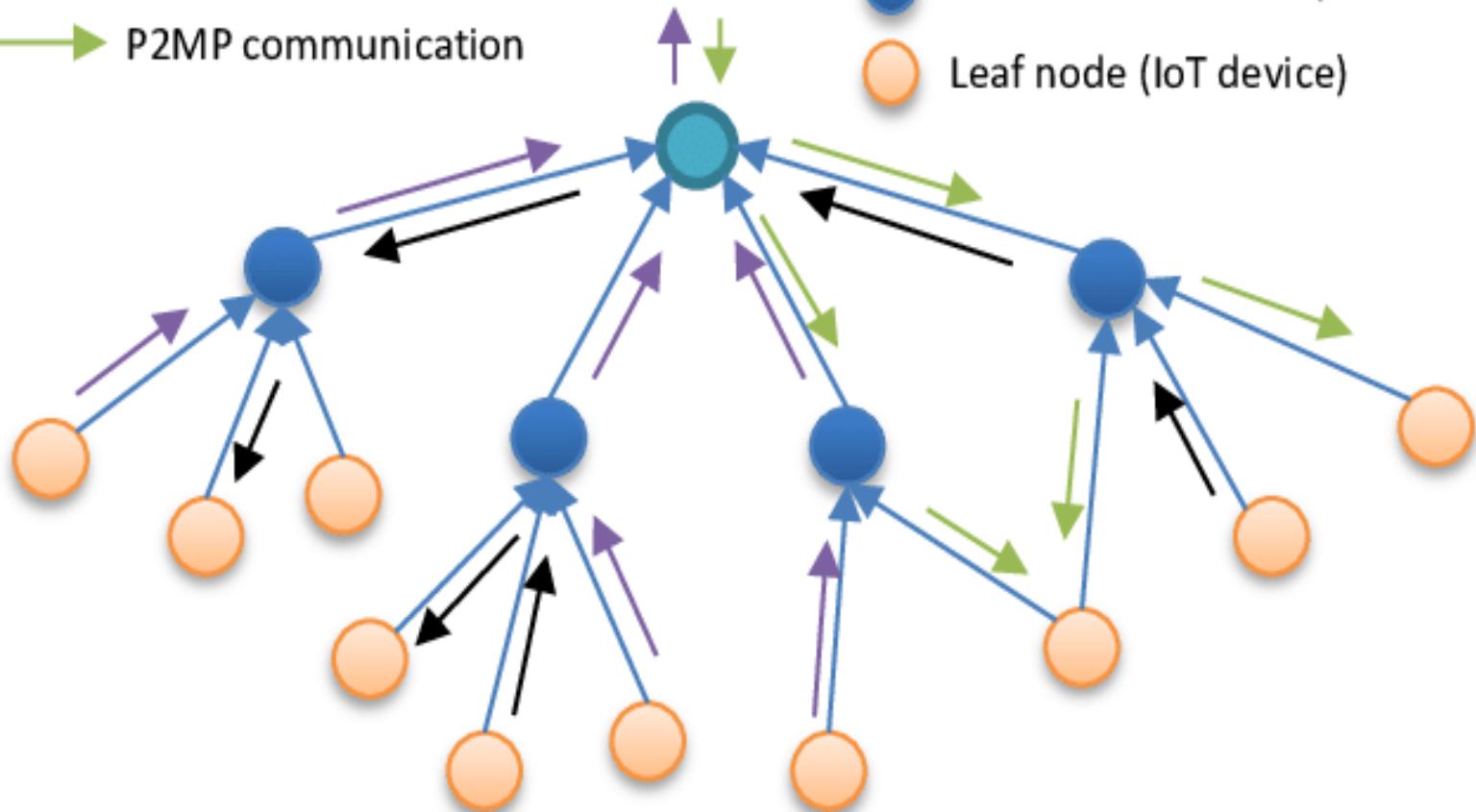
RPL

- RPL is a
 - Distance vector routing protocol and
 - Source routing protocol
- RPL forms a tree called DODAG (Destination Oriented Directed Acyclic Graph) with two varieties of DODAG (Grounded or Floating).
- Grounded DODAG: In which nodes send their traffic to the gateway node and the gateway will forward them to the destination on behalf of them.
- Floating DODAG: It has no gateway node and each node is responsible to forward its traffic.

Example DODAG

- P2P communication
- MP2P communication
- P2MP communication

- DODAG root (Gateway)
- Intermediate nodes (IoT device)
- Leaf node (IoT device)



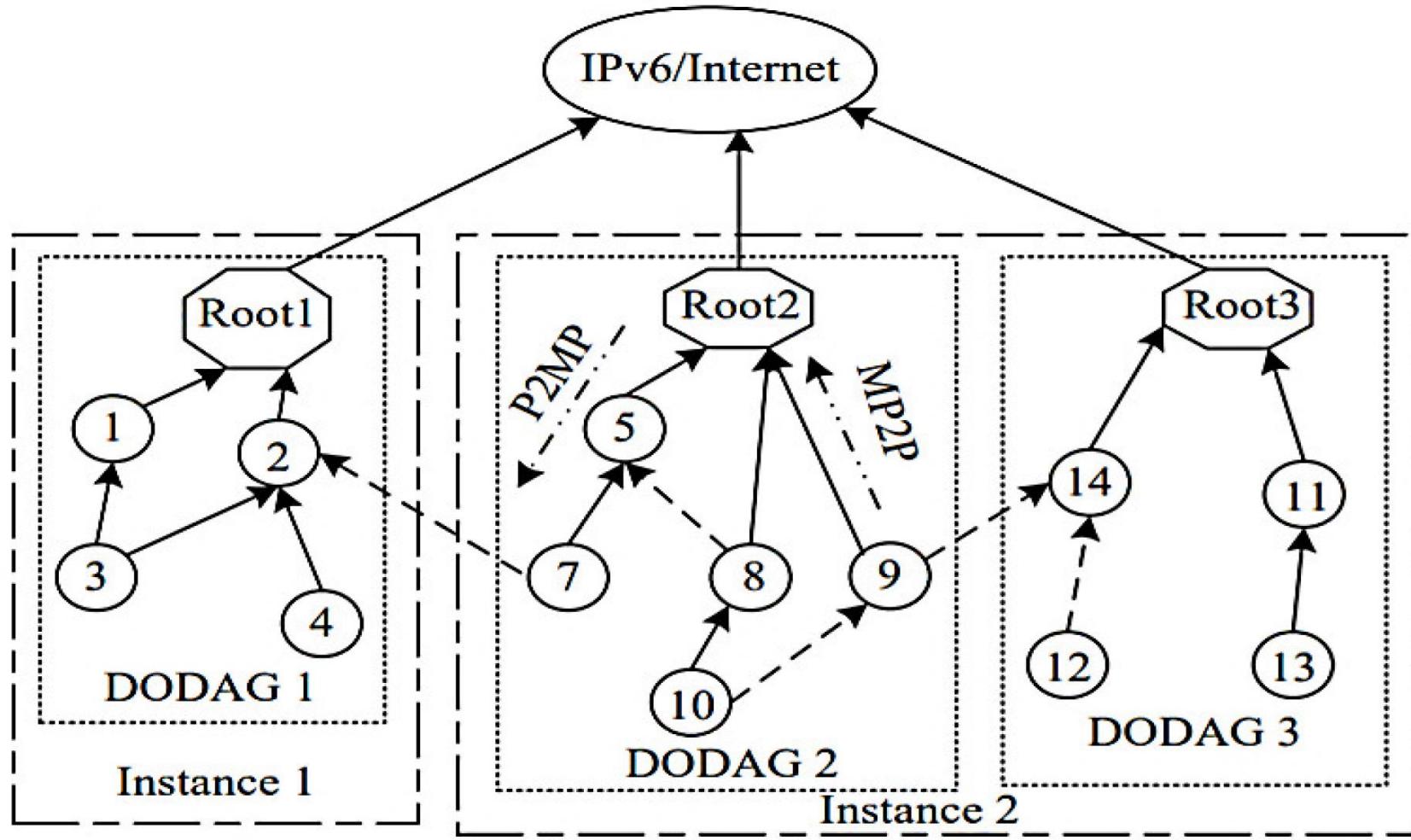
RPL

- Each node has a rank value that is calculated with respect to the gateway node according to a predefined cost metric like hop count, bandwidth, reliability or number of transmissions.
- Each node in an RPL network has a preferred parent which works as the gateway of this node to the destination.
- If an RPL node didn't find any path in its routing table for a packet, the node forwards the packet to its preferred parent and so on until it either reaches the destination or a common parent which forwards it down the tree towards the destination.
- Nodes in an RPL network must have routes for all the nodes down the tree. It means that the nodes which are near to the root node must have large routing table entries.

RPL Topology

- An RPL DODAG is a DAG graph rooted and pointed at a single destination.
- A DODAG graph is identified uniquely using a combination of DODAG ID and RPL Instance ID.
- the RPL instance can contain more than one DODAG each one of them has an DODAG ID.
- Each node in the DODAG has a rank value which expresses the node position with respect to the DODAG root node.
- Rank values strictly increase towards down direction and decrease in the up direction as it becomes closer to the root node.
- DODAG Root is responsible for the routes aggregation and DODAG construction.
- Traffic towards the root node is considered as Multi_point to Point (MP2P) while the traffic from root node to leaves nodes considered as Point to Multi_point (P2MP).

DODAG ID and Instance ID



RPL Instance

- A RPL Node may belong to multiple RPL Instances, and it may act as router in some and as a leaf in others.
- Type: Local and Global
- Control and Data packets have RPL Instance field.

Local RPL Instance

- Is always a single DODAG whose singular root owns the corresponding DODAGID and allocates the local RPL InstanceID.
- D=0 in control messages
- D is used in data packets to indicate whether the DODAGID is the source or Destination of the packet. D=1 the dest.
- Address of the packet must be the DODAGID.



Local RPLInstanceID in 0...63

Global RPL Instance

- Are coordinated, have one or more DODAGs, and are typically long-lived.
- A global RPL InstanceID must be unique to the whole LLN.

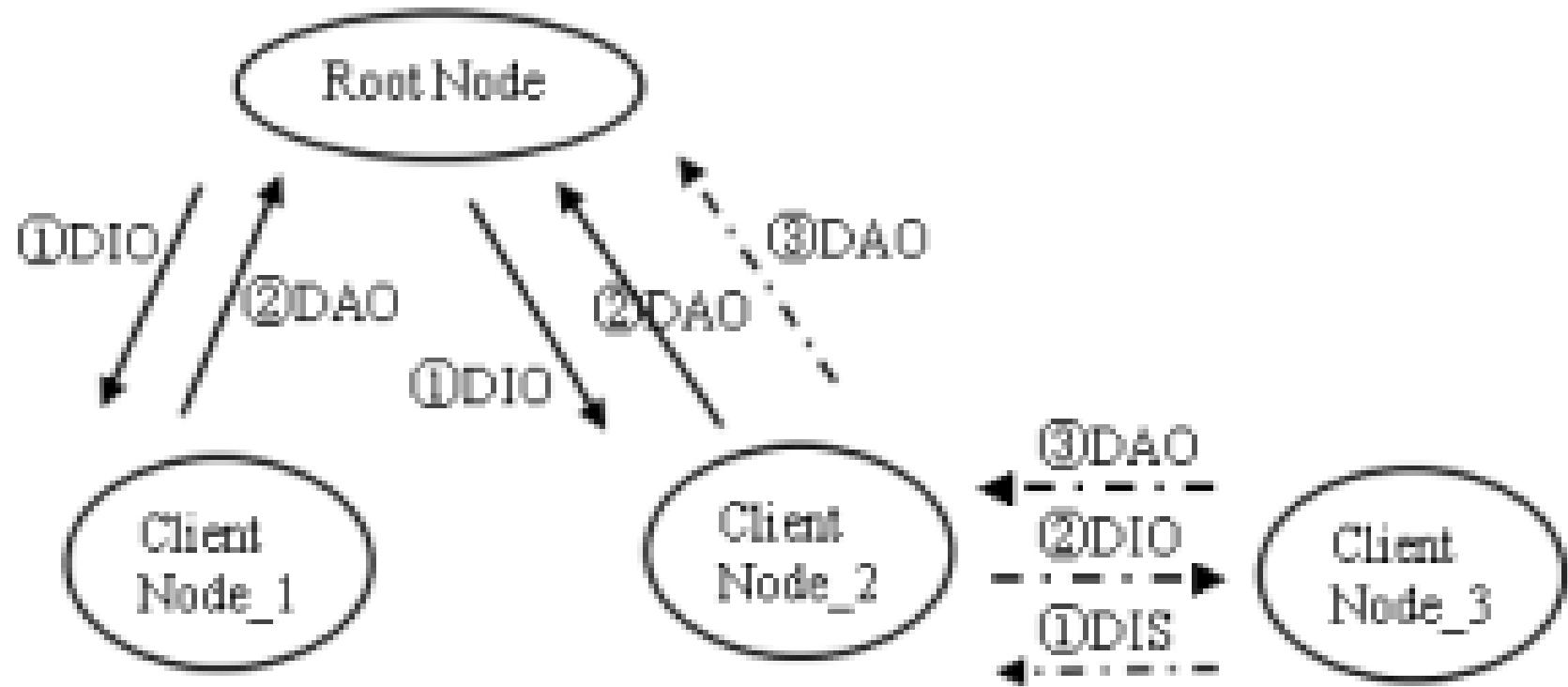


Global RPLInstanceID in 0...127

DODAG control messages

- RPL protocol has four types of ICMPv6 control messages.
- The control messages are:
 - **DODAG information object (DIO)**
 - **Destination advertisement object (DAO)**
 - **DODAG information solicitation (DIS)**
 - **DAO acknowledgment (DAO-ACK)**

DODAG Control Messages



DODAG control Messages are ICMP

Type=155	Code	Checksum
	Base	
	Option(s)	

Code: Identify the type of control message

0x00 → DODAG Information Solicitation (DIS)

0x01 → DODAG Information Object (DIO)

0x02 → Destination Advertisement Object (DAO)

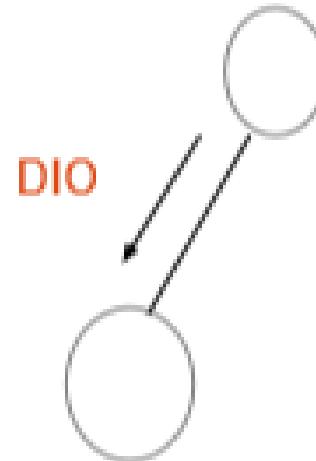
0x03 → DAO-ACK

DODAG information object : DIO

- The DODAG root (border router node) issues DIO message in a multicast form to construct a new DODAG.
- Used to keep the current rank /level of the node
- Determine the distance of each node to the root
- Choose the preferred parent path
- The DIO message structure contains all information concerning the network that allows any node to find an RPL instance, select a DODAG parent set, learn about its configuration parameters, and finally build the DODAG.

DIO

- Carries information that allows a node to:
 - Discover a RPL instance
 - Learn its configuration parameters
 - Select a DODAG parent set
 - Maintain the DODAG



DIO

RPLInstanceID				Version Number	Rank	
G	0	MOP	Prf	DTSN	Flags	Reserved
DODAGID						
Option(s)						

DIO Base Object

DAO

- As the DODAG is being constructed each node in the DODAG sends this message to propagate and populate a node rank and routing tables' information to their predecessor nodes that support the downward route traffic (traffic towards leaves nodes).
- Helps RPL to maintain upward and downward traffic.
- The Destination Advertisement Object (DAO) message is used to propagate reverse route information to record the nodes visited along the upward path.

DAO

- The exact way Rank is computed depending on the DAG's Objective Function (OF).
- The Rank may analogously track a simple topological distance, may be calculated as a function of link metrics, and may consider other properties such as constraints.
- The DAO message may optionally, upon explicit request or error, be acknowledged by its destination with a Destination Advertisement Acknowledgement (DAO-ACK) message back to the sender of the DAO.

DAO

RPLInstanceID	K	D	Flags	Reserved	DAOSequence
DODAGID					
Option(s)					

DAO Base Object

RPLInstanceID	D	Reserved	DAOSequence	Status
DODAGID				
Option(s)				

DAO-ACK Base Object

DIS

- Used by a specific node in order to acquire DIO messages from another reachable adjacent node.
- These messages are sent by any node to trigger the others to send DIO messages to this node and this happened only when that node didn't receive a correct DIO message for a long time.

DAO-ACK

- Used as a response to a DAO msg and is sent by a DAO recipient node like a DAO parent or DODAG root.

DODAG Creation

- A DODAG will start its formation when the root node starts sending its location information using DIO message to all low-power lossy network (LLN) levels.
- The process of the DODAG construction begins from the DODAG root.
- First, the DODAG root broadcasts the DODAG information by transmitting a DIO message.
- The DIO contains the DODAGID (Used to identify the root node and it's corresponding DODAG.)
- The DIO message is processed and transmitted to other nodes one by one.

DODAG creation

- Neighbors of the root receive and process the DIO message.
- The DIO message is processed and transmitted to other nodes one by one.
- A node which has not joined any DODAGs calculates the cost of the path to the DIO sender and then decides whether to join the DODAG or not.
- Once a neighbour joins the DODAG, it has a route towards the DODAG root, and the root becomes a DODAG parent of the node.

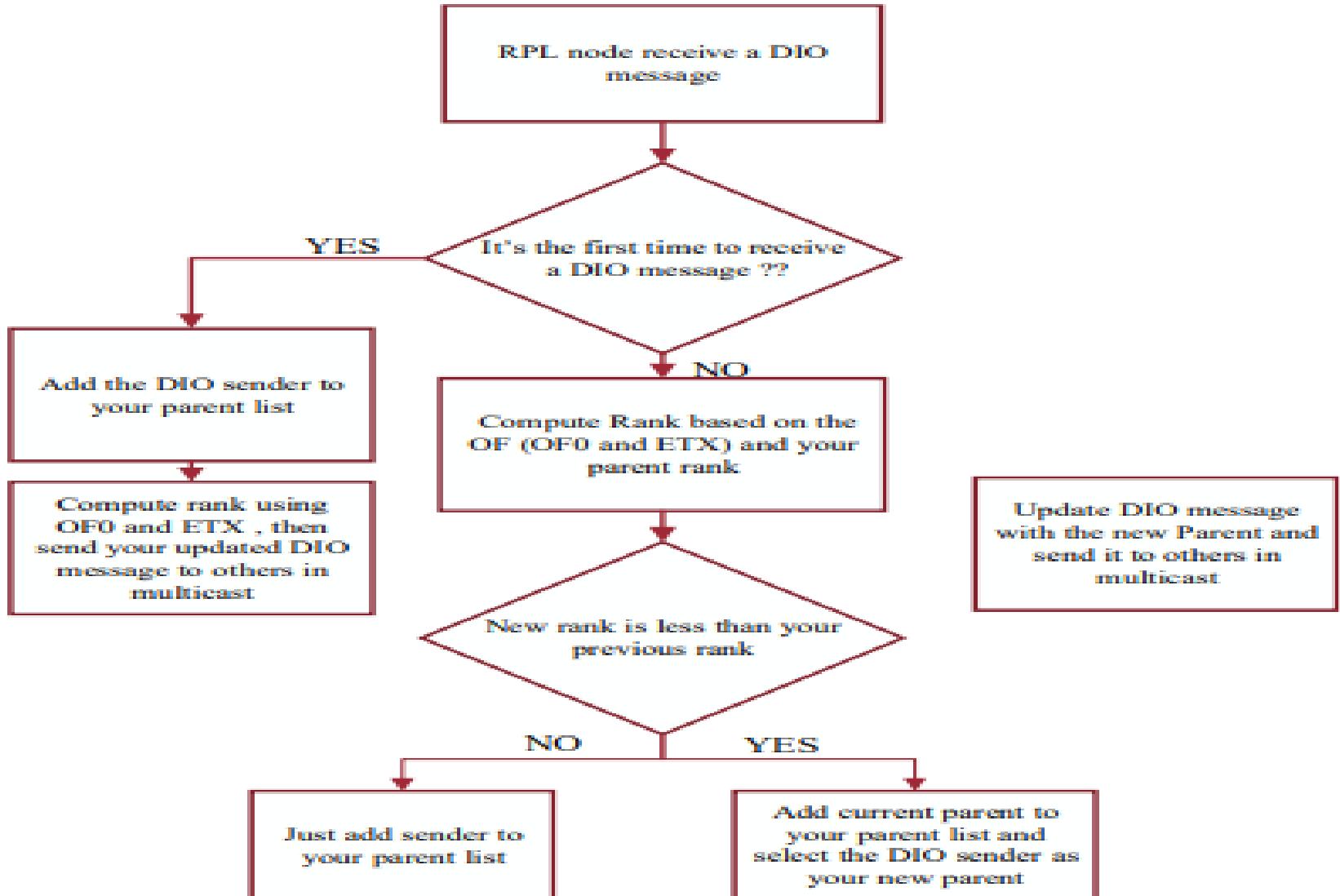
DODAG creation

- Next, the node calculates its Rank in the DODAG and replies with a DAO message to its parent to inform its participation.
- A node which has not received any DIO messages and has not joined any DODAGs can request DODAG information by sending DIS messages periodically to its neighbours.
- All of the neighbours repeat this process until all of the nodes join the DODAG.

Points to remember

- The node's rank must be greater than its parents rank
- Each node chooses the most preferred parent and forwards the DIO message in a multicast manner with the new rank information to the other nodes. When an RPL node receives a DIO message, it can do one of three things:
 - Analyze the received DIO message to decide if it will maintain its existing location or change it to another lower depending on the path cost and the Objective Function proposed within the DODAG.
 - To avoid occurrence of routing loops, a node must drop all nodes in its parent's list with ranks lower than the new calculated node's rank. After the DODAG has been constructed, each node would have a defined route formed by the most preferred parents to the destination root node,

RPL DIO message mechanism

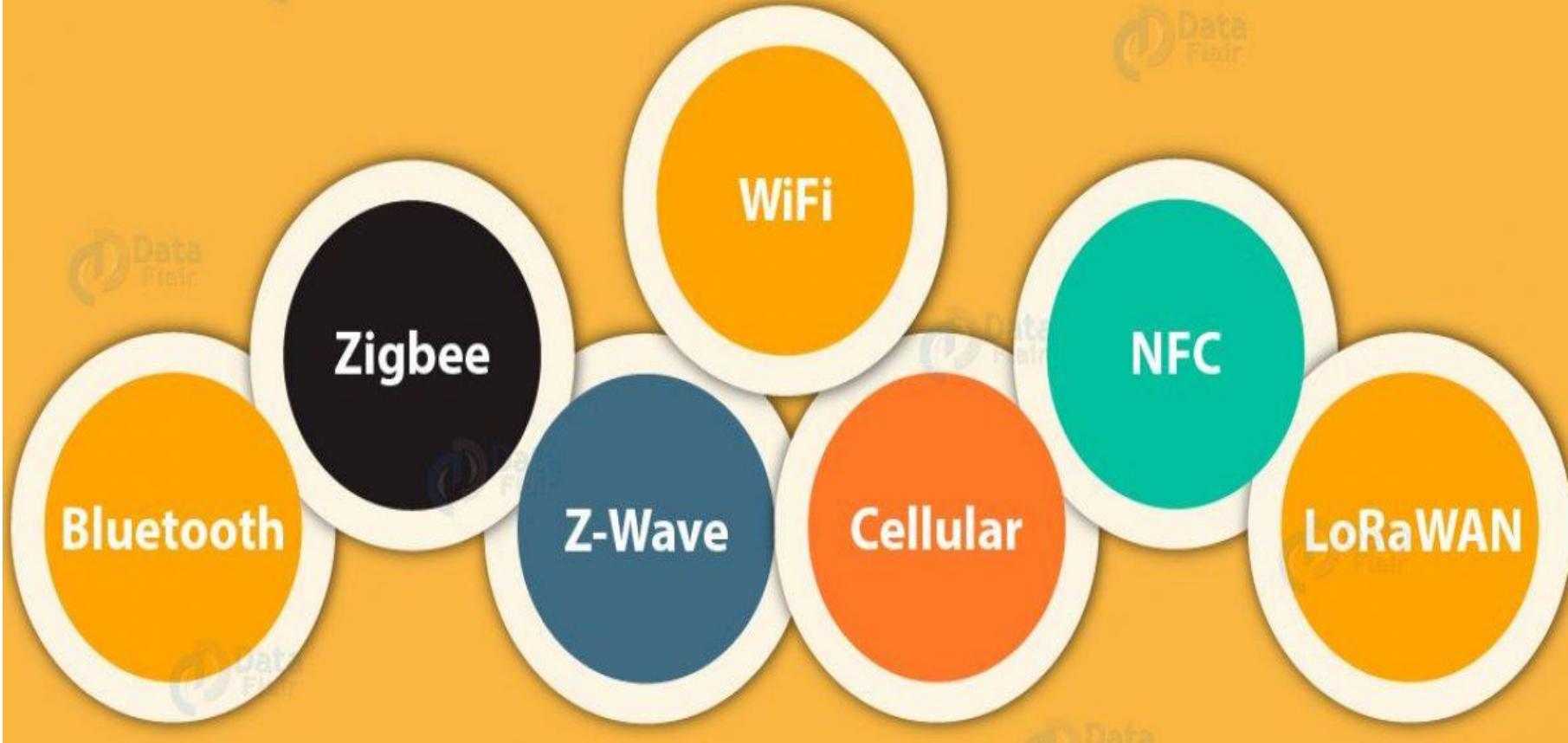


Internet of Things: Zigbee / IEEE802.15.4

Lect #20

**Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008**

IOT Technologies and Protocols



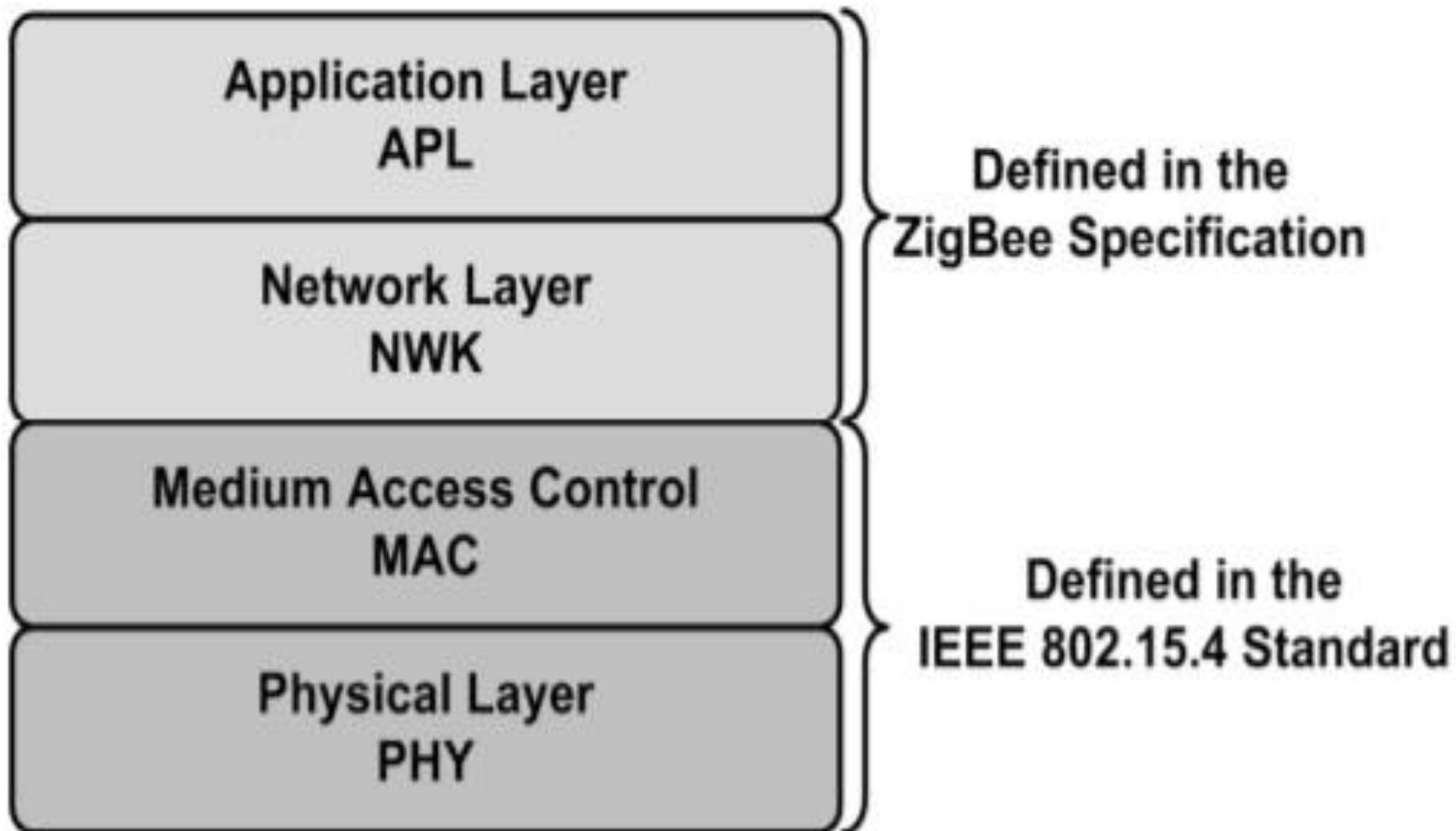
Infrastructure Protocols: IEEE 802.15.4

- The IEEE 802.15.4 protocol specifies the Medium Access Control (MAC) sub-layer and physical layer for Low-Rate Wireless Personal Area Networks (LR-WPAN).
- The IEEE 802.15.4 protocol targets low data rate, low power consumption, low cost wireless networking, which typically fits the requirements of sensor networks.

Zigbee / IEEE 802.15.4

- The ZibBee Alliance, has been working in conjunction with IEEE (task group 4) in order to specify a full protocol stack for low cost, low power, low data rate wireless communications, as well as to foster its use worldwide.
- The ZigBee Specification, released in December 2004, specifies the protocol layers above IEEE 802.15.4, i.e. the network (including security services) and the application (including device objects and profiles) layers.

Zigbee / IEEE 802.15.4 protocol stack architecture



Network devices of IEEE 802.15.4

- According to the IEEE 802.15.4 standard, a LR-WPAN supports two different types of devices:
 - Full Function Device (FFD)
 - Reduced Function Device (RFD)

Full Function Device (FFD)

- a FFD is a device that can support three operation modes, serving as:
 - **A Personal Area Network (PAN) Coordinator**: the principal controller of the PAN. This device identifies its own network, to which other devices may be associated.
 - **A Coordinator**: provides synchronization services through the transmission of beacons. Such a coordinator must be associated to a PAN coordinator and does not create its own network.
 - **A simple device**: Rest of the normal devices.

Reduced Function Device

- Reduced Function Device (RFD):
- The RFD is a device operating with minimal implementation of the IEEE 802.15.4 protocol.
- It is intended for applications that are extremely simple, such as a light switch or a passive infrared sensor.
- They do not have the need to send large amounts of data and may only associate with a single FFD at a time.

LR-WPAN

- A LR-WPAN must include at least one FFD acting as a PAN coordinator that provides global synchronization services to the network and manages potential FFDs and RFDs.

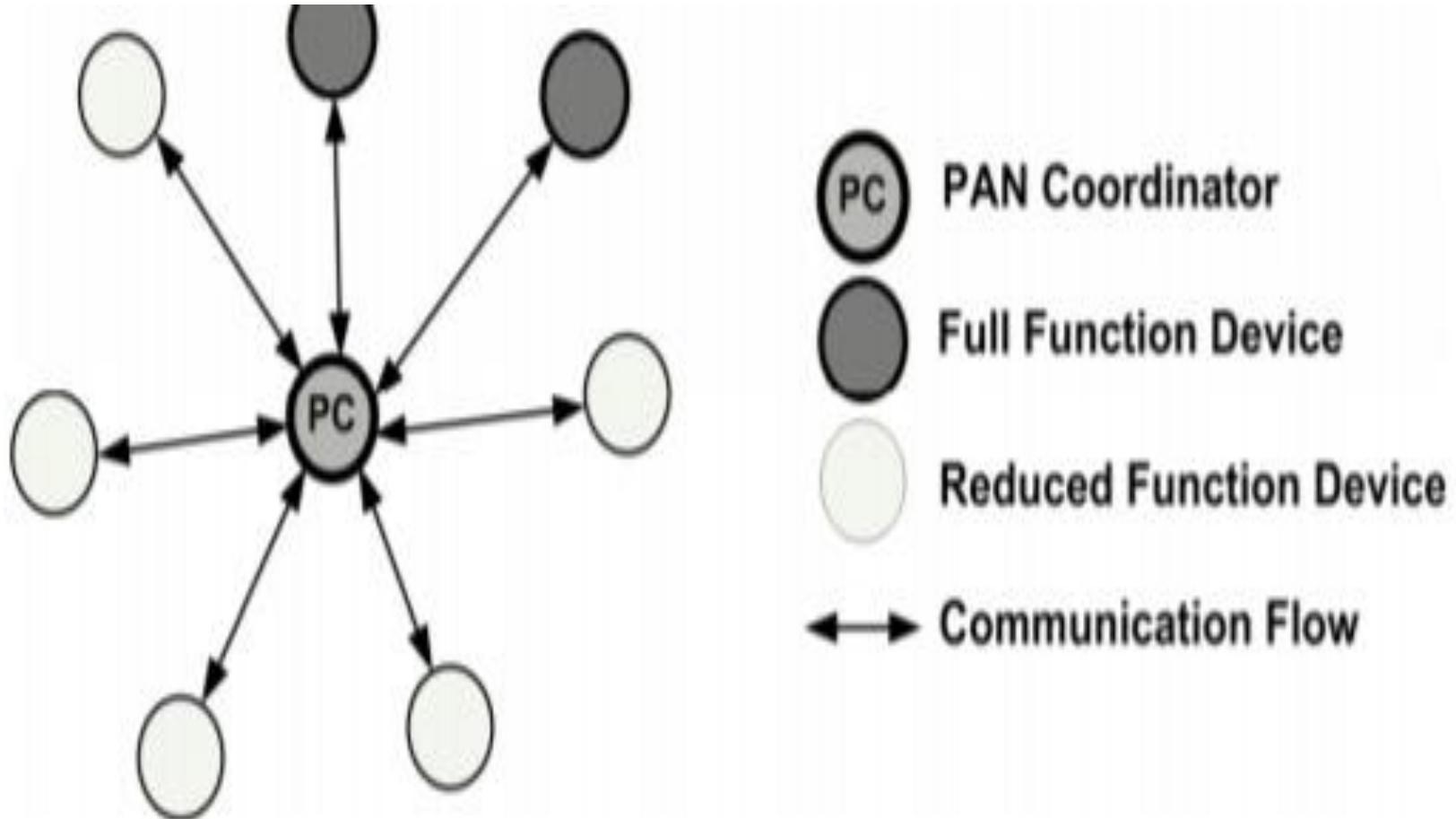
IEEE 802.15.4 Network Topologies

- Two basic types of network topologies are defined in the IEEE 802.15.4 standard.
 - **the star topology**
 - **peer-to-peer topology.**
 - **cluster-tree topology** (special case of peer-to-peer topology.)

IEEE 802.15.4 star topology

- In the star topology, a unique node operates as a PAN coordinator.
- When the FFD is activated it may establish its own network and become its PAN coordinator.
- The PAN coordinator chooses a PAN identifier, which is not currently used by any other network.
- Star topology is used in home automation, personal computer peripherals, toys and games.

Star Topology



Star Topology

- The communication paradigm in the star topology is centralized i.e., each device (FFD or RFD) joining the network and willing to communicate with other devices must send its data to the PAN coordinator, which dispatch them to the destination devices.
- Due to the power-consuming tasks of the PAN coordinator in the star topology, the IEEE 802.15.4 standard mentions that the PAN coordinator may be mains powered while other devices are more likely to be battery powered.

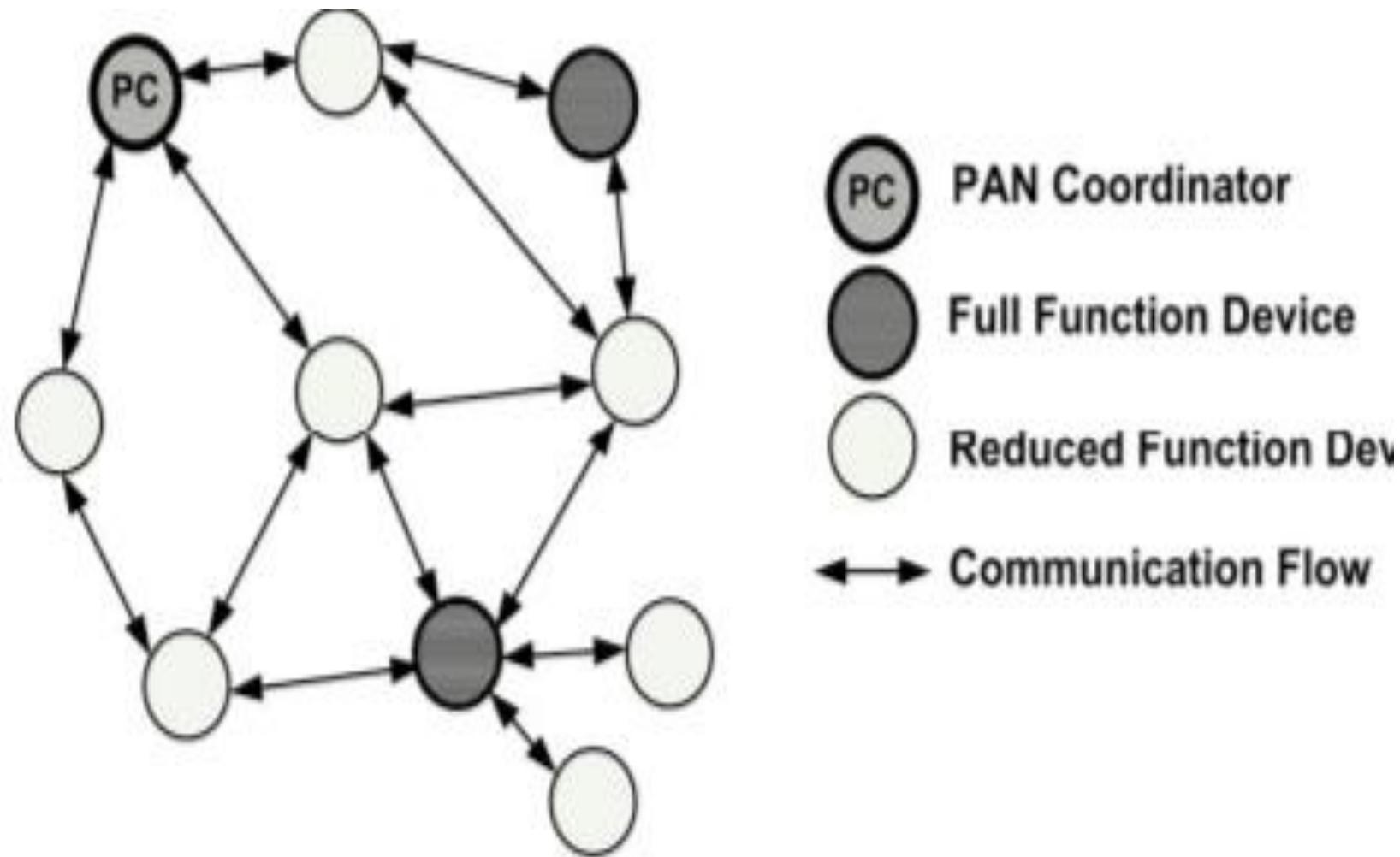
Flaws of Star Topology

- As a consequence, the star topology seems to be not adequate for traditional wireless sensor networks, since all sensor nodes are supposed to be battery-powered and thus very energy-constrained.
- A sensor node selected as a PAN coordinator will get its battery resources rapidly ruined.
- A potential bypass to this problem is to have a dynamic PAN coordinator based on remained battery supplies in sensor nodes.

IEEE 802.15.4 Peer-to-peer Topology

- The peer-to-peer topology also includes a PAN coordinator, which is nominated, for instance, by virtue of being the first device to communicate on the channel.
- However, the communication paradigm in the peer-to-peer topology is decentralized, where each device can directly communicate with any other device in its radio range.

Peer-to-peer topology model



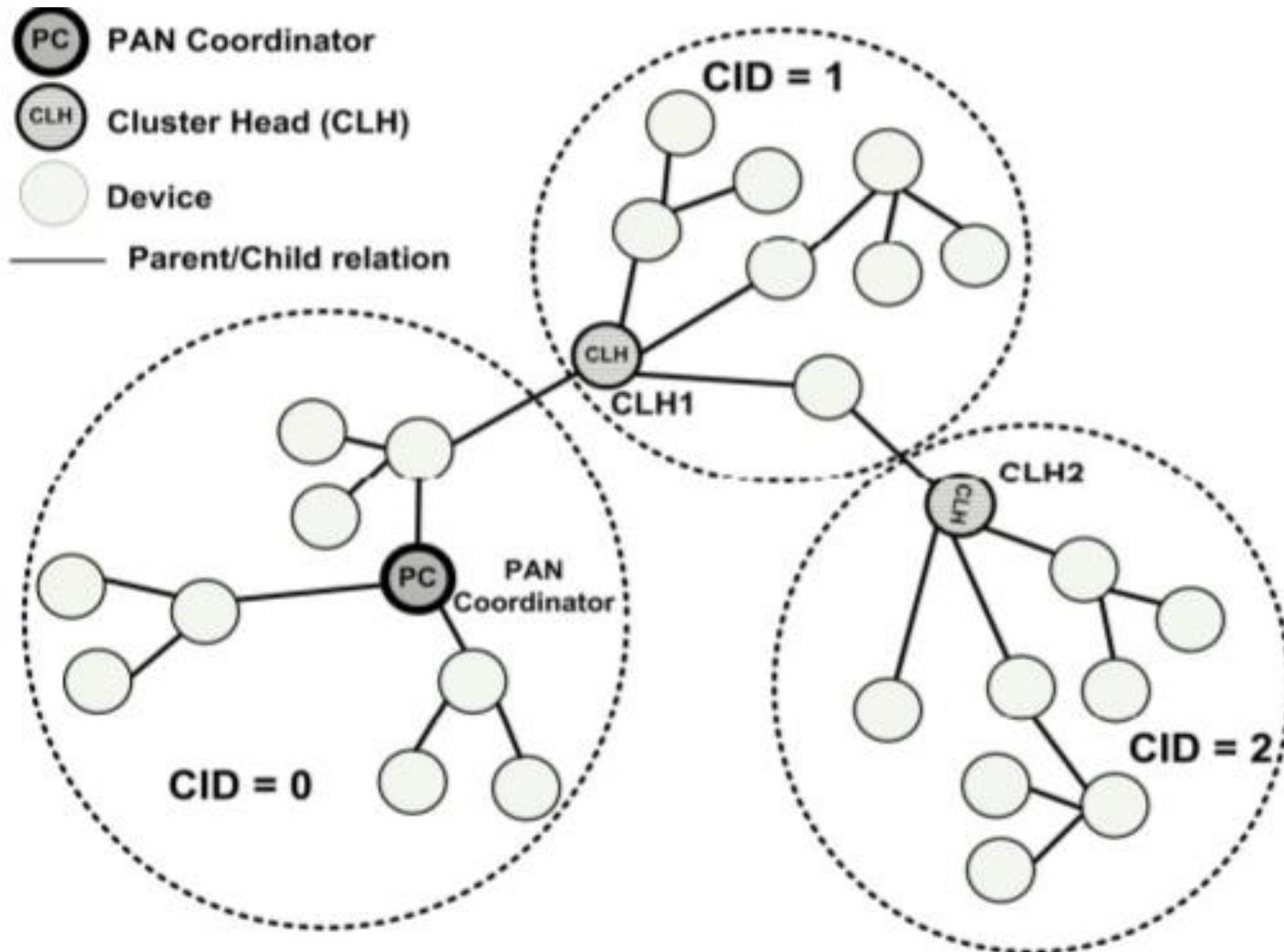
Peer –to-Peer Topology

- This mesh topology enables enhanced networking flexibility, but it induces an additional complexity for providing an end-to-end connectivity between all devices in the network.
- Basically, the peer-to-peer topology operates in ad hoc fashion and allows multiple hops to route data from any device to any other device.
- Wireless Sensor Networks are one of the applications of such a topology. The resource usage is fairer in the peer-to-peer topology since the communication process does not rely on a particular node.

IEEE 802.15.4 : The cluster Tree

- The Cluster-Tree is a special case of a peer-to-peer network in which most devices are FFDs.
- One (and only one) coordinator is nominated as the PAN coordinator, which identifies the entire network.
- Any FFD may act as a coordinator and provide synchronization services to other devices or other coordinators.
- An RFD connects to a cluster-tree as a leaf node at the end of a branch and associates itself with only one FFD.

Cluster tree topology



Cluster Formation

- **A. The PAN coordinator ?**
 - forms the first cluster by establishing itself as Cluster Head (CLH) with a cluster identifier (CID) equals to zero ?
 - chooses an unused PAN identifier, ?
 - Broadcasts beacons to neighbouring devices

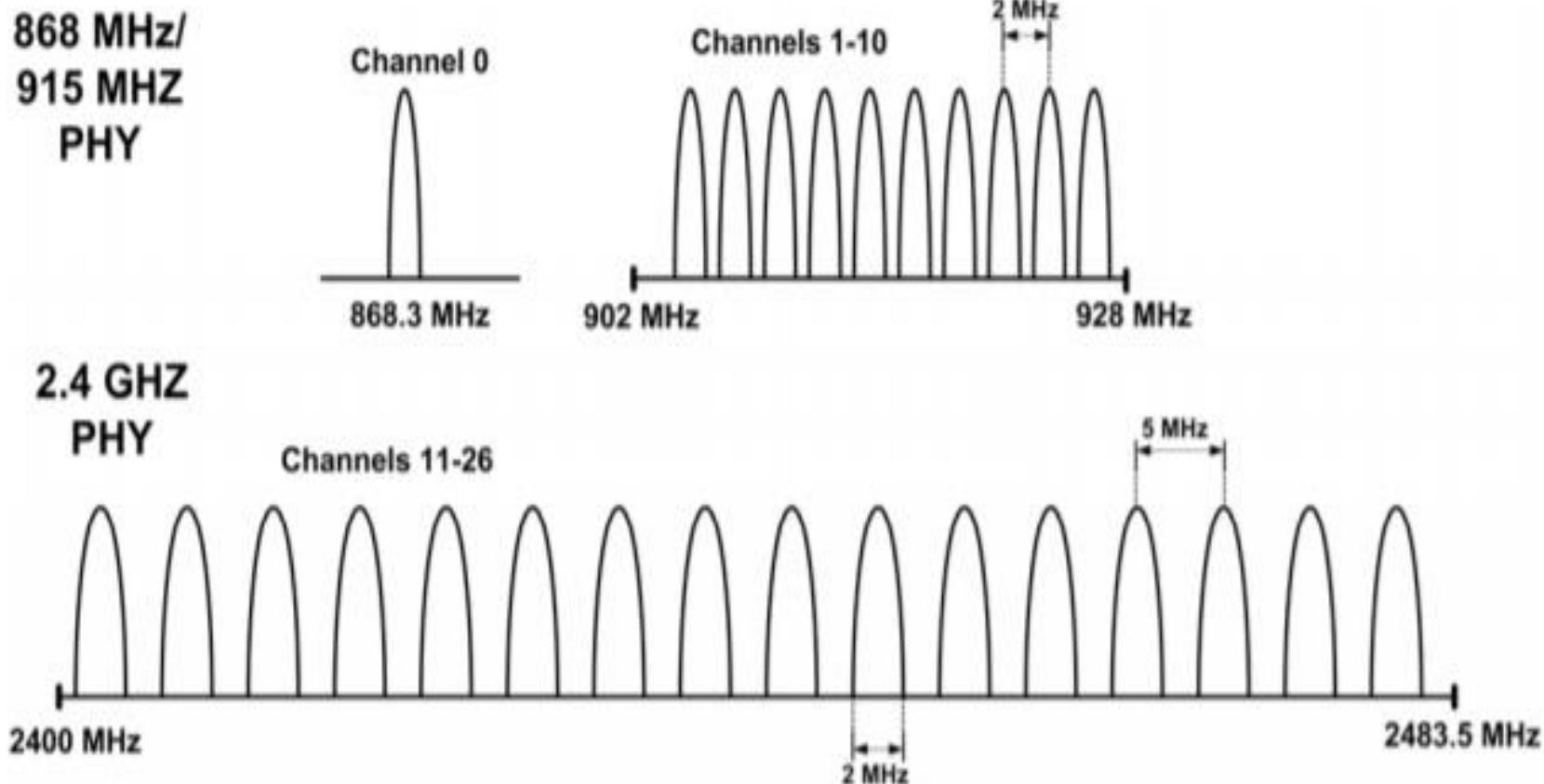
Cluster Formation

- **B. A candidate device receiving a beacon frame may request to join the network to the CLH**
 - If the PAN accepts the request to join the network, it adds the candidate device as a child device in its neighbour list. In turn, the new joined device adds the CLH as its parent in its neighbour list and starts transmitting periodic beacons. Other devices hearing these beacons may join the network at this device.
 - If for some reason the candidate device cannot join the network at the cluster head, it will search for another parent device.

IEEE 802.15.4 Physical Layer

- The physical layer is responsible for data transmission and reception using a certain radio channel and according to a specific modulation and spreading technique.
- The IEEE 802.15.4 offers three operational frequency bands: 2.4 GHz, 915 MHz and 868 MHz.
- There is a single channel between 868 and 868.6 MHz, 10 channels between 902 and 928 MHz, and 16 channels between 2.4 and 2.4835 GHz.
- The data rate is 250 kbps at 2.4 GHz, 40 kbps at 915 MHz and 20 kbps at 868 MHz.

Operating Frequency Bands



Operating Frequency Bands

- Lower frequencies are more suitable for longer transmission ranges due to lower propagation losses.
- Low rate transmissions provide better sensitivity and larger coverage area.
- Higher rate means higher throughput, lower latency or lower duty cycles.

Internet of Things:

IEEE802.15.4

Lect #21

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-
769008

Physical Layer Tasks

- The physical layer of the IEEE 802.15.4 is in charge of the following tasks:
 - Activation and deactivation of the radio transceiver
 - Energy Detection (ED) within the current channel
 - Link Quality Indication (LQI)
 - Clear Channel Assessment (CCA)
 - Channel Frequency Selection

Activation and deactivation of the radio transceiver

- The radio transceiver may operate in one of three states: **transmitting, receiving or sleeping.**
- Upon the request of the MAC sub-layer, the radio is turned ON or OFF.
- The turnaround time from transmitting to receiving and vice versa should be no more than 12 symbol periods according to the standard (each symbol corresponds to 4 bits).

Energy Detection within the current channel

- It is an estimation of the received signal power within the bandwidth of an IEEE 802.15.4 channel.
- The energy detection time should be equal to 8 symbol periods.
- This measurement is typically used by the network layer as a part of channel selection algorithm or for the purpose of Clear Channel Assessment (CCA), to determine if the channel is busy or idle.

Link Quality Indication

- The LQI measurement characterizes the Strength/Quality of a received packet.
- It measures the quality of a received signal on a link.
- This measurement may be implemented using receiver ED, a signal to noise estimation or a combination of both techniques.
- The LQI result may be used by the higher layers (Network and Application layers).

Clear Channel Assessment

- This operation is responsible for reporting the medium activity state: busy or idle. The CCA is performed in three operational modes:
 - Energy Detection mode: the CCA reports a busy medium if the detected energy is above the ED threshold.
 - Carrier Sense mode: the CCA reports a busy medium only if it detects a signal with the modulation and the spreading characteristics of IEEE 802.15.4 and which may be higher or lower than the ED threshold.

Clear Channel Assessment

- Carrier Sense with Energy Detection mode: The CCA reports that the medium is busy only if it detects a signal with the modulation and the spreading characteristics of IEEE 802.15.4 and with energy above the ED threshold.

IEEE 802.15.4 MAC

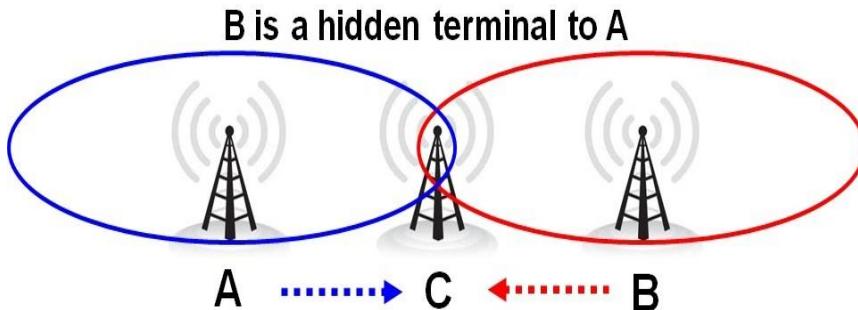
- The MAC sub-layer of the IEEE 802.15.4 protocol provides an interface between the physical layer and the higher layer protocols of LR-WPANs.
- The MAC sub-layer of the IEEE 802.15.4 protocol has many common features with the MAC sub-layer of the IEEE 802.11 protocol, such as the use of CSMA/CA (Carrier Sense Multiple Access / Collision Avoidance) as a channel access protocol, the support of contention-free and contention-based periods.

IEEE 802.15.4 MAC

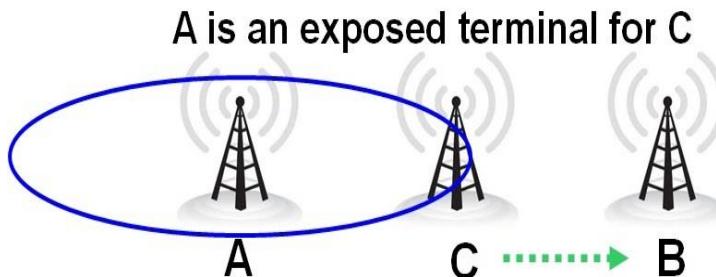
- However, the specification of the IEEE 802.15.4 MAC sub-layer is adapted to the requirements of LR-WPAN as, for instance, eliminating the RTS/CTS mechanism (used in IEEE 802.11) to reduce the probability of collisions, since collisions are more likely to occur in low rate networks.

Challenges of wireless networks

- Hidden Terminal

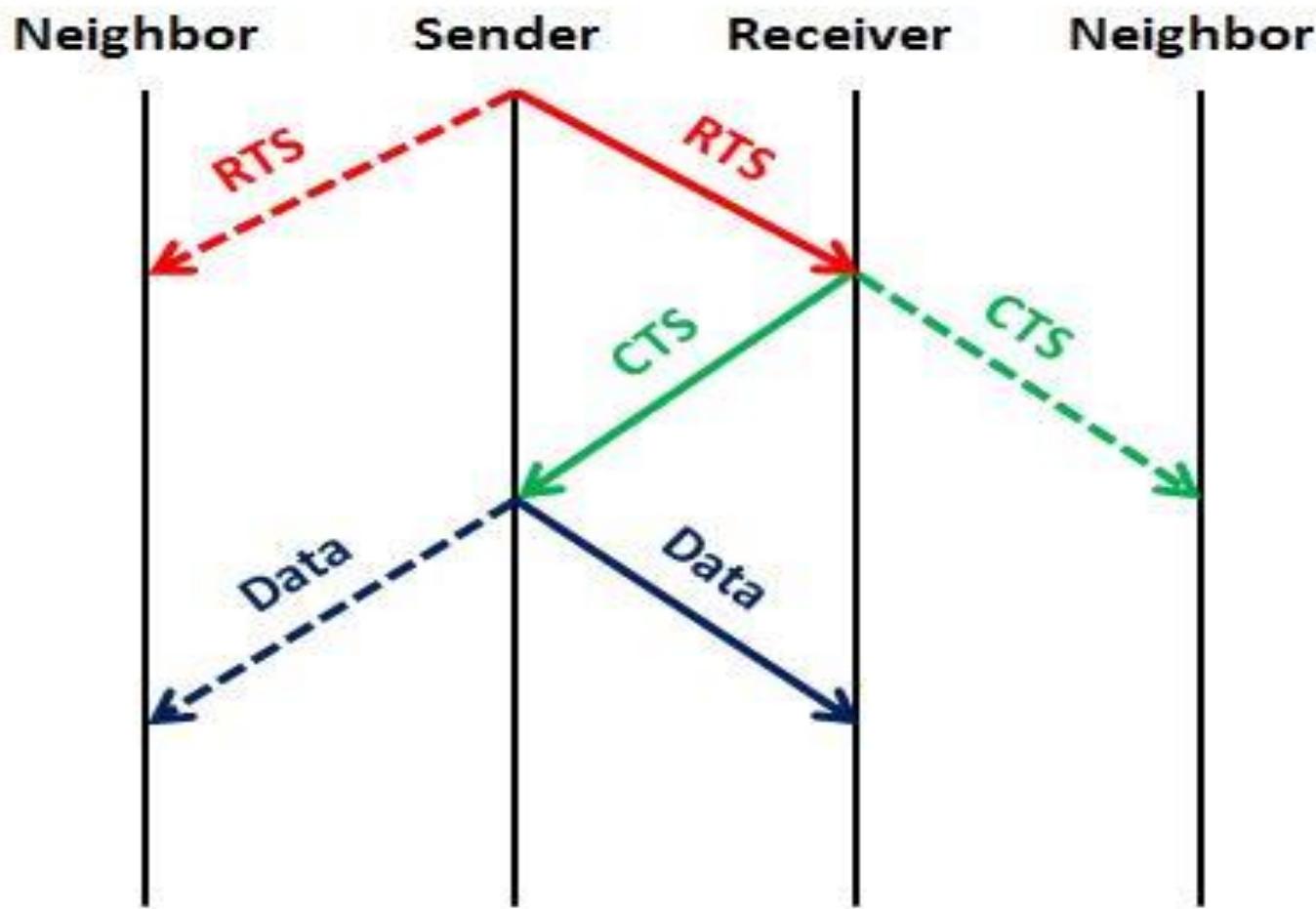


- Exposed Terminal

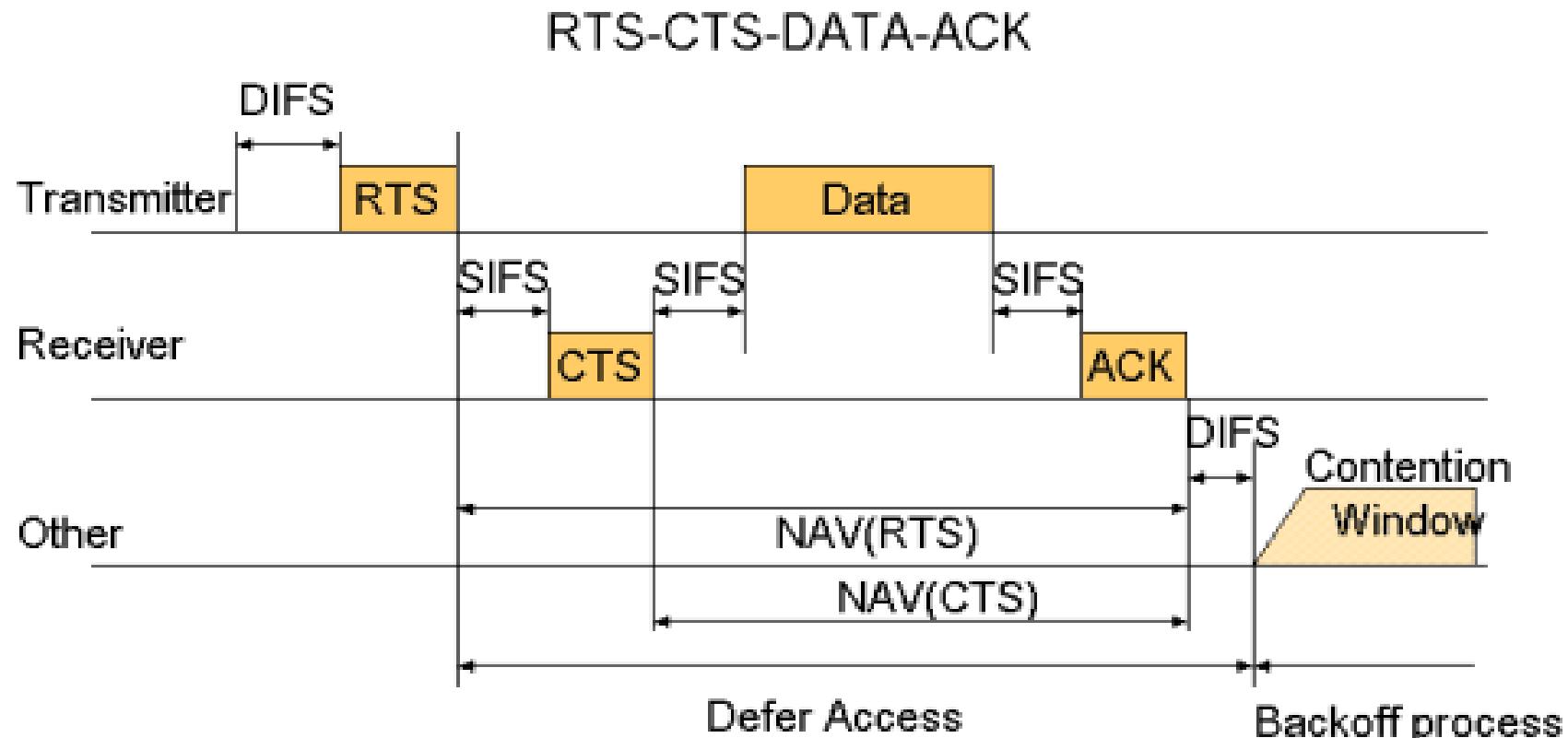


- Solution????
 - RTS (Request to Send)
 - CTS (Clear to send)

Solution..



IEEE 802.11: Network Allocation Vector (NAV)



DIFS: Distributed IFS

RTS: Request To Send

SIFS: Short IFS

CTS: Clear To Send

ACK: Acknowledgement

NAV: Network Allocation Vector

DCF: Distributed Coordination Function

Classifications of 802.11 MAC

- Contention based
- Contention based with reservation mechanism
- Contention based with scheduling mechanism

IEEE 802.11: Example - Contention Based

C is not replying.

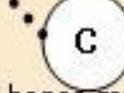
Congestion may be? I
backoff... ⊕ (10)



RTS (8)

I can listen RTS but can
NOT reply. (9)

Now I can reply to D
for old RTS with
RRTS (12)



I heard CTS from B hence my
transmission is deferred till
data is to be received (4)



CTS (3) →
RTS (1) ←
DS (5) →



Date received. Sending ACK (11)

RTS heard from A. Hence
my transmission is
deferred till CTS should be
received (2)

I have heard now CTS.
Hence my transmission is
deferred till data is to be
received (4)

Internet of Things

F is not replying.
Congestion may be? I
backoff... ⊕ (10)



RTS (8) ·

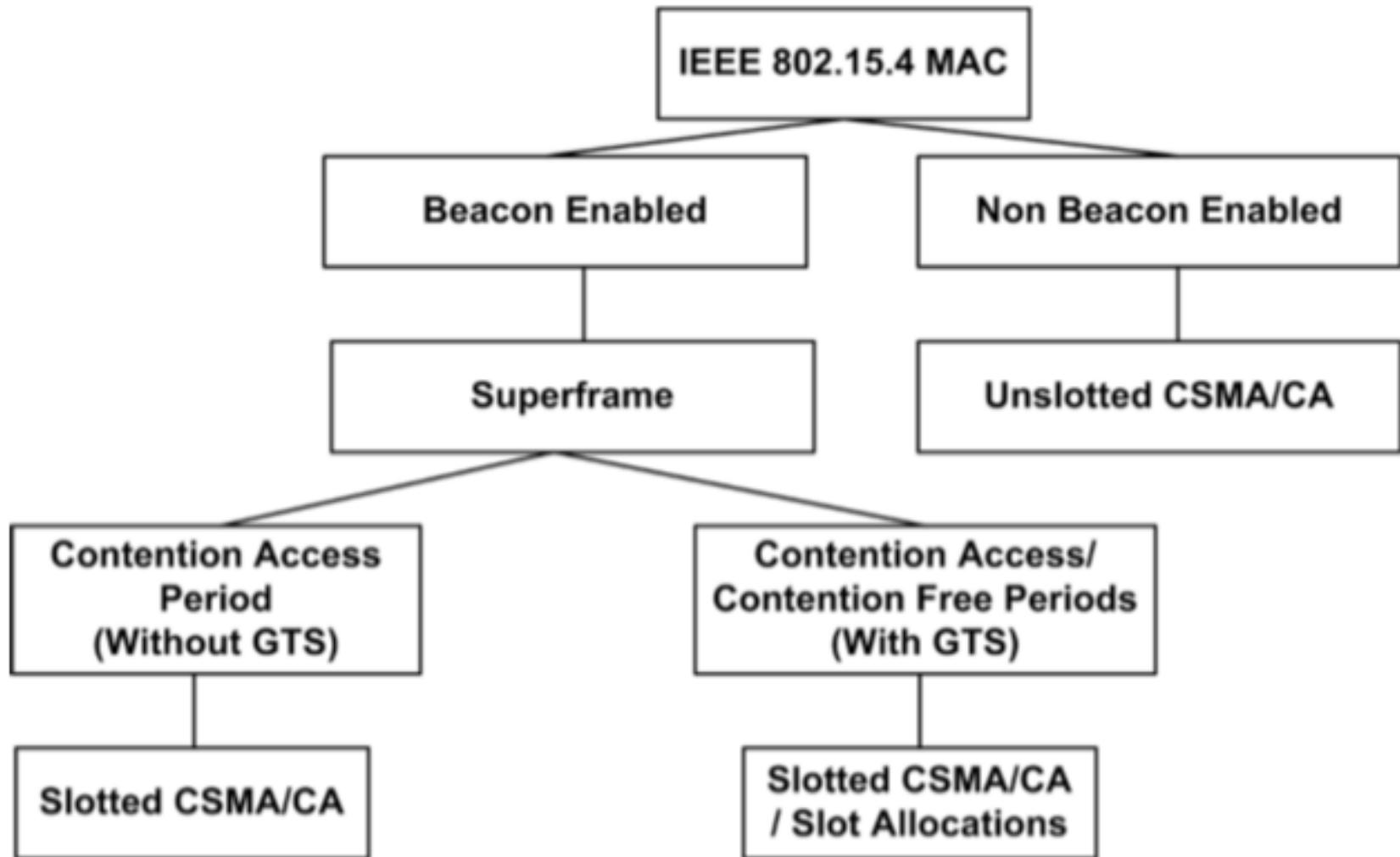


I can NOT listen any
thing any more. ⊕ (9)

DS tells me that what is
data length and RTS/CTS
exchange a success. I
defer my transmissions
till data ends. (6)

An Example about working of MACAW

IEEE 802.15.4 operational modes



IEEE 802.15.4 operational Mode

- The Beacon-enabled Mode
- The Non Beacon enabled mode

The Beacon-enabled mode

- Beacons are periodically generated by the coordinator to synchronize attached devices and to identify the PAN.
- A beacon frame is (the first) part of a super frame, which also embeds all data frames exchanged between the nodes and the PAN coordinator.
- Data transmissions between nodes are also allowed during the superframe duration.

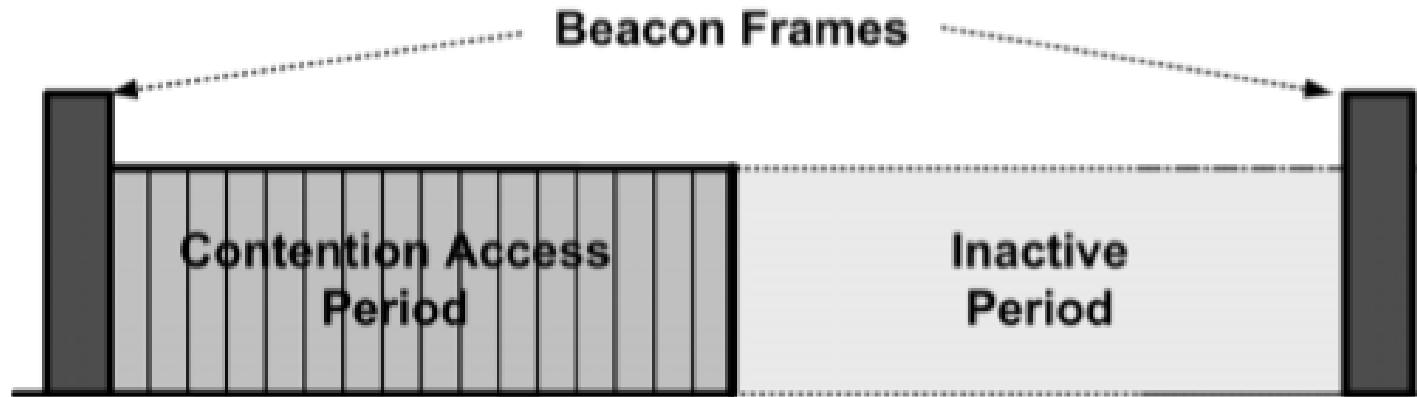
The Beacon-enabled mode

- When the coordinator selects the beacon-enabled mode, it forces the use of a superframe structure to manage communication between devices (that are associated to that PAN).
- The format of the superframe is defined by the PAN coordinator and transmitted to other devices inside every beacon frame, which is broadcasted periodically by the PAN coordinator.

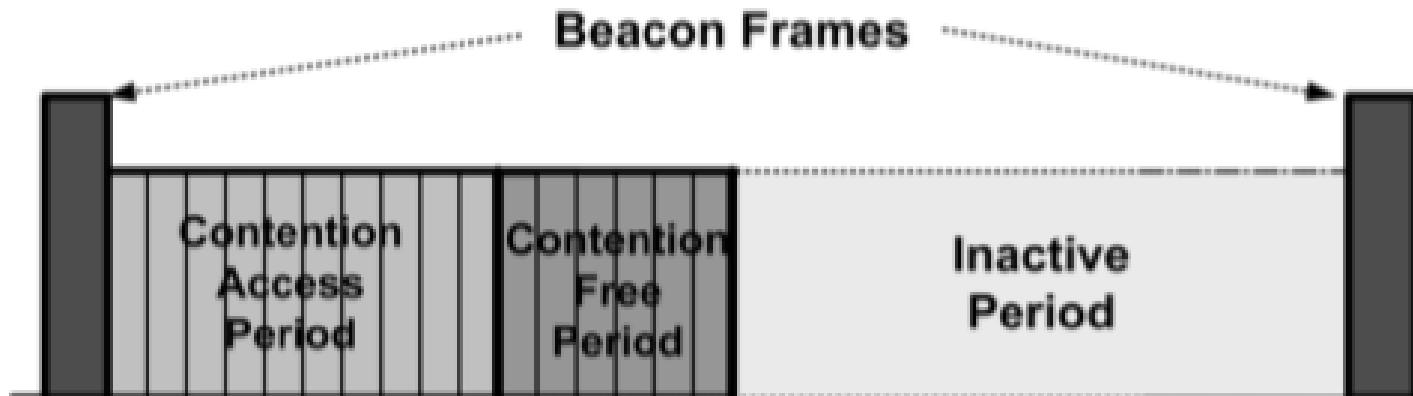
The Beacon-enabled mode

- The superframe is divided into 16 equally sized slots and is followed by a predefined inactive period.
- the superframe is contained in a Beacon Interval, which is bounded by two consecutive beacon frames, and includes one Contention Access Period (CAP) and may include also a Contention Free Period (CFP).

Superframes



A. Superframe structure without guaranteed time slots (GTSs)



B. Superframe structure with GTSs

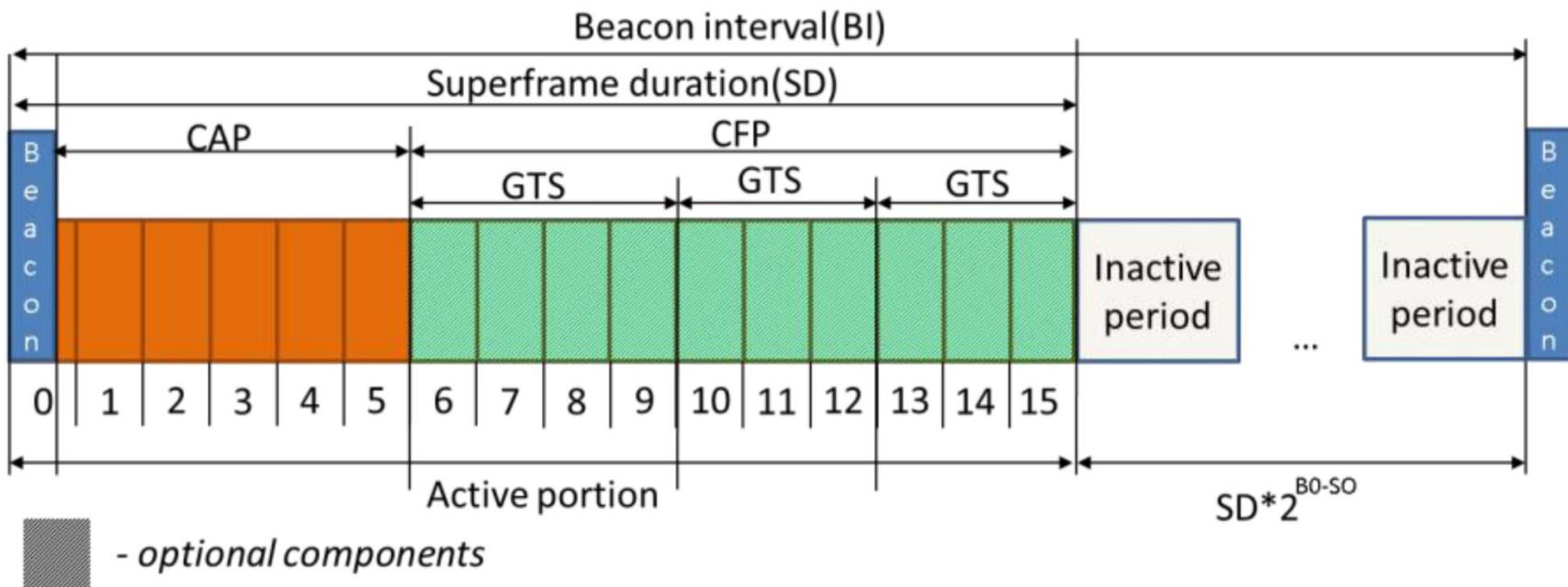
Superframe structure without GTS

- If communications are restricted to the CAP (defined in the beacon, issued by the PAN Coordinator) a device wishing to communicate must compete with other devices using a slotted CSMA/CA mechanism.
- All transmissions must be finished before the end of the superframe, i.e., before the beginning of the inactive period (if exists).

Superframe structure with GTS

- If some guaranteed QoS is to be supported, then a Contention-Free Period (CFP) is defined.
- The CFP consists in Guaranteed Time Slots (GTSs) that may be allocated by the PAN coordinator to applications requiring low-latency or specific data bandwidth requirements.
- The CFP is a part of the superframe and starts at a slot boundary immediately following the CAP
- The PAN coordinator may allocate up to seven GTSs and each GTS may occupy more than one time slot.

Example



Superframe structure with GTS

- With this superframe configuration, all contention-based communication must be finished before the start of the CFP, and a node transmitting a GTS must ensure that its transmission will be complete before the start of the next GTS (or the end of the CFP).
- According to the standard, the GTS is used only for communications between a PAN coordinator and a device.

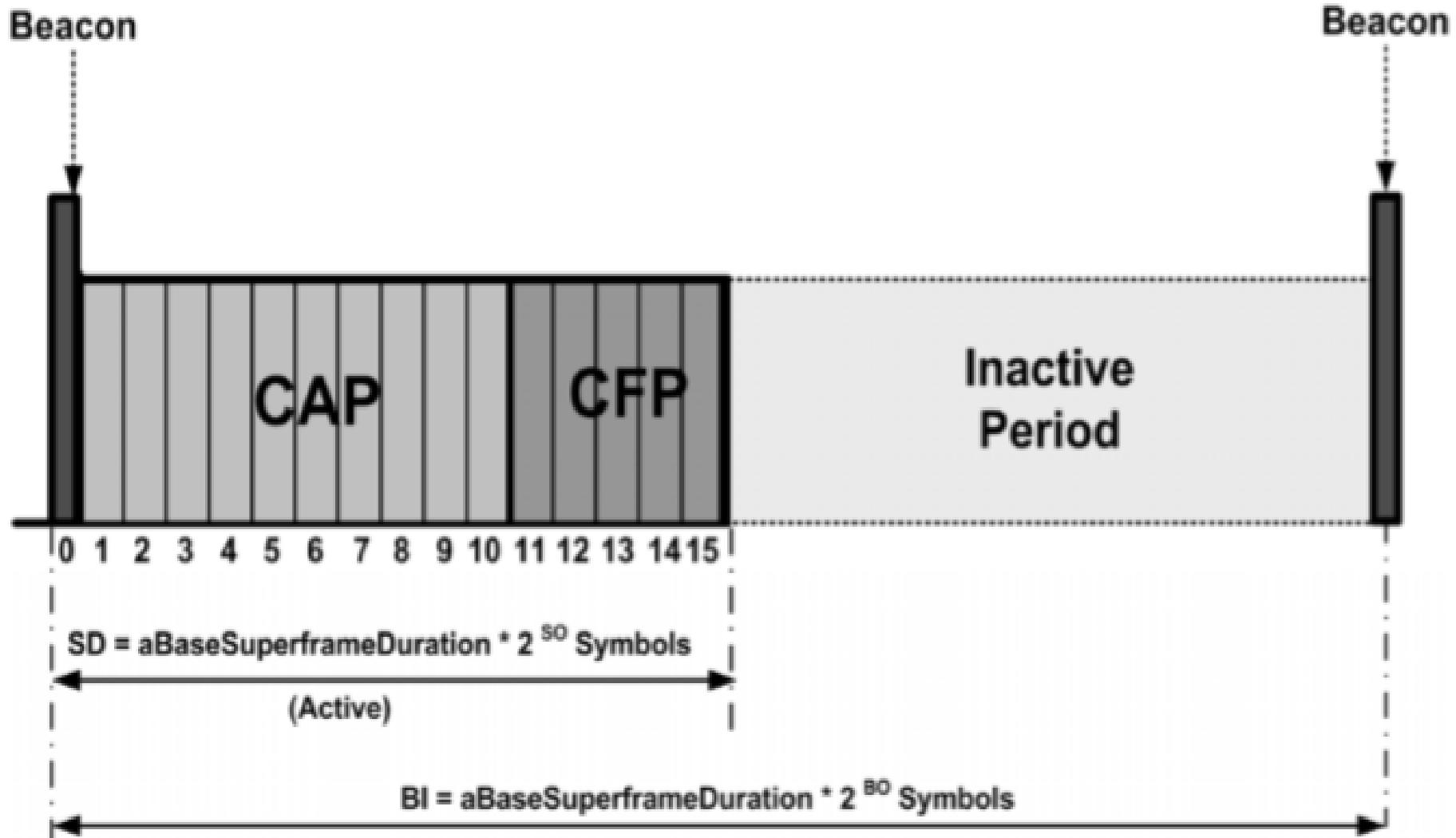
Inactive Period

- In both configurations (CAP only or CAP/CFP), the superframe structure can have an inactive period during which the PAN coordinator does not interact with its PAN and may enter in a low power mode.
- Switching the network between activity/inactivity periods is very suitable for devices where reduced energy consumption is a main concern.
- In fact, the inactive periods enable the devices to save energy and thus extend network lifetime.

The Non Beacon-enabled Mode

- When the PAN coordinator selects the non-beacon enabled mode, there are neither beacons nor superframes.
- Medium access control is provided by an unslotted CSMA/CA mechanism.

The Superframe structure



The Superframe Structure

- The superframe is contained in a Beacon Interval bounded by two beacon frames, and has an active period and an inactive period.
- The PAN coordinator interacts with its PAN during the active period, and enters in a low power mode (sleep) during the inactive period.

The Superframe Structure

- The structure of a superframe is defined by two parameters:
- macBeaconOrder (BO): this attribute describes the interval at which the coordinator must transmit beacon frames. The value of the macBeaconOrder and the Beacon Interval (BI) are related as

for $0 \leq BO \leq 14$,

$$BI = aBaseSuperframeDuration * 2^{BO} \text{ symbols}$$

The Superframe Structure

- Mac Superframe Order (SO): this attribute describes the length of the active portion of the superframe, which includes the beacon frame. The value of the mac Superframe Order and the Superframe Duration (SD) are related as

for $0 \leq SO \leq BO \leq 14$,

$$SD = aBaseSuperframeDuration * 2^{SO} \text{ symbols}$$

The Superframe Structure

- If $SO = BO \Rightarrow SD = BI$ then the superframe is always active.
- According to the standard, if $SO = 15$, the superframe will not be active following the beacon.
- Moreover, if $BO = 15$, then the superframe shall not exist and the network will operate in the non beacon-enabled mode.
- In this case, the value of SO is ignored.

The Superframe Structure

- As a result, a PAN that wishes to use the superframe structure must set mac Beacon Order to a value between 0 and 14 and mac Superframe Order to a value between 0 and the value of mac Beacon Order.
- Otherwise, the PAN will operate in a non beacon-enabled mode with a value of mac Beacon Order and mac Superframe Order equal to 15.

The Superframe Structure

- The active portion of each superframe is divided into $aNumSuperframeSlots=16$ equally spaced slots of $2^{SO} * aBaseSlotDuration$ duration.
- The attribute $aBaseSlotDuration$ represents the number of symbols forming a superframe slot when the superframe order is equal to zero.
- The value of $aBaseSlotDuration$ is equal to 60 symbols.

Active portion of superframe

- The active portion of the superframe structure is composed of three parts:
- **Beacon:** the beacon is transmitted without the use of CSMA at the start of slot 0. It contains the information on the addressing fields, the superframe specification, the GTS fields, the pending address fields, etc.

Active portion of superframe

- **CAP:** the CAP starts immediately after the beacon frame and ends before the beginning of the CFP (if it exists). Otherwise, the CAP ends at the end of the active part of the superframe.

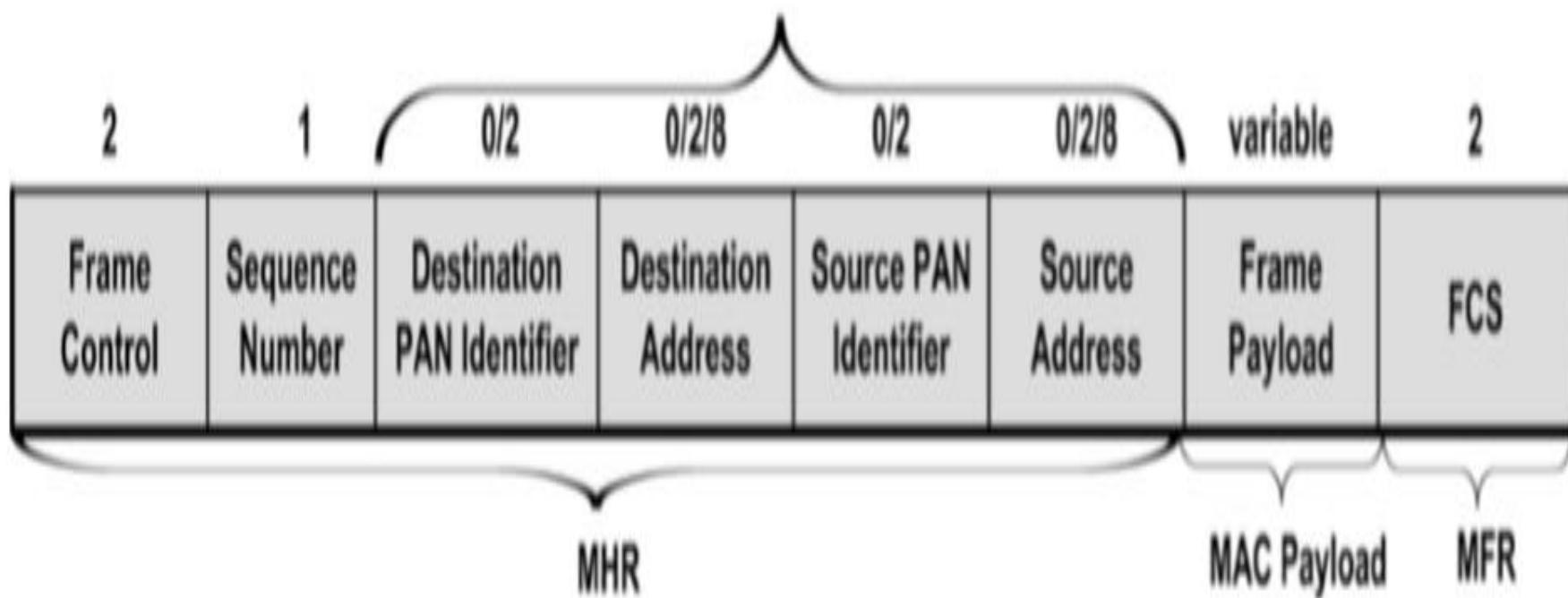
Active portion of superframe

- **CFP:** The CFP starts immediately after the end of the CAP and must complete before the start of the next beacon frame. All the GTSs that may be allocated by the PAN coordinator are located in the CFP and must occupy contiguous slots. The CFP may therefore grow or shrink depending on the total length of all GTSs. The transmissions in the CFP are contention-free and therefore do not use a CSMA/CA mechanism to access the channel.

IEEE 802.15.4 Frame Formats

- The standard defines four frame formats
 - MAC frames,
 - beacon frames,
 - the data frames
 - and acknowledgment frames

MAC Frame Formats



Description: MAC Frames

- Each MAC frame consists of the following basic three parts:
 - MAC Header (MHR)
 - MAC Payload
 - MAC Footer (MFR)

MAC Header (MHR)

- It includes:
 - **Frame Control**: it is a 16-bits field and contains information defining the frame type and other control flags (Security Enabled, Frame Pending, Acknowledgment Request, Intra-PAN ...).
 - **Sequence Number**: it is an 8-bit field and specifies a unique sequence identifier for the frame.
 - **Destination PAN Identifier**: it is a 16-bit field that specifies the unique PAN identifier of the intended recipient of the frame

MAC Header (MHR) Contd..

- **Destination Address:** it is either a 16-bit or 64bit field (depending on the value of the destination addressing subfield of the Frame Control field) that specifies the address of the intended recipient of the frame.
- **Source PAN Identifier:** it is a 16-bit field that specifies the unique PAN identifier of the originator of the frame.
- **Source Address:** it is either a 16-bit or 64-bit field (depending on the value of the destination addressing subfield of the Frame Control field) that specifies the address of the originator of the frame.

MAC Payload

- The MAC Payload of variable length, which contains information specific to individual frame types.

MAC Footer (MFR)

- The MAC Footer (MFR), which contains the Frame Check Sequence (FCS) field. The FCS field is 16 bits in length and contains a 16 bit Cyclic Redundancy Check (CRC).

Internet of Things: LPWAN

Autumn 2020-21

Lecture # 22

Prof. Suchismita Chinara
Dept. of Computer Science Engg.
National Institute of Technology Rourkela-769008
Email: suchismita@nitrkl.ac.in

Low Power Wide Area Networking

- Low power wide area networking technologies (LPWAN) technologies are used extensively for communication in the IoT ecosystem.
- LPWAN technologies are superior when compared to Bluetooth and BLE for M2M communication because of their cost effectiveness and low-power consumption.
- LPWAN technology is ideal for connecting devices that send **small amounts of data over a long range with battery efficiency**.

Low Power Wide Area Networking

- An ideal example of devices that fall under this category are sensors, which are used for transmitting data within smart homes, buildings, parking systems, and so on.
- LPWANs can accommodate packet sizes from 10 to 1,000 bytes at uplink speeds of upto 200 Kbps. LPWAN's long range varies from 2 km to 1,000 km, depending on the technology.

Low Power Wide Area Networking

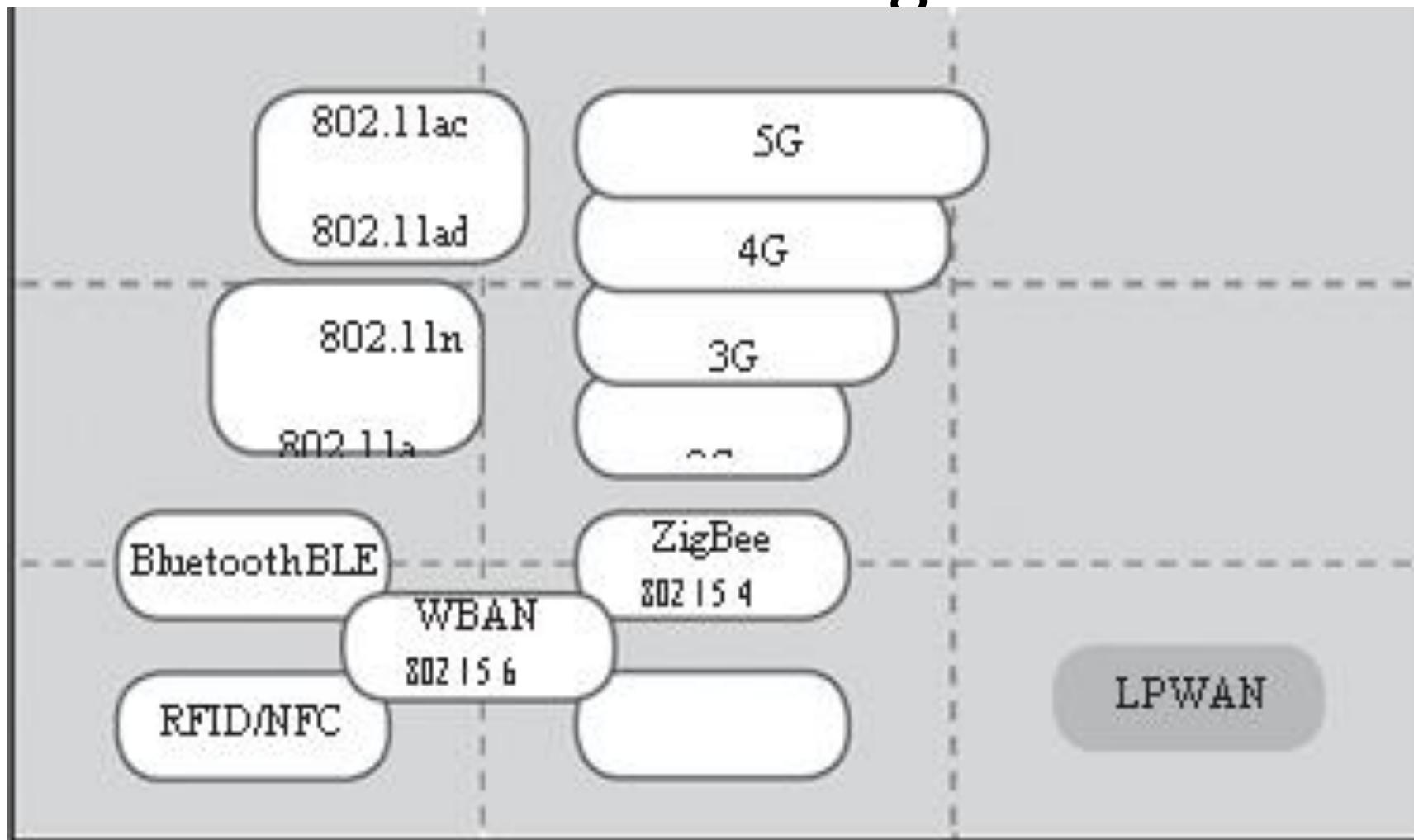
- The key features of LPWAN that make it suitable for IoT ecosystem are:
 - *Long-range communication*: Ability to support nodes that are \geq 10 km distance from the gateway. However, the correct distance is based on the LPWAN technology that is used.
 - *Low transmission data rate*: Less than 5000 bits of data are sent per second. Often only 20–256 bytes per message sent several times a day.
 - *Low-power consumption*: This provides very long battery life for the devices. Many times, the battery life may last up to 10 years.

Low Power Wide Area Networking

- LPWAN technology is ideally suited for the following two types of applications:
 - *Fixed, medium-to-high density connections:* This is mainly used in cities and for buildings as an alternative option for cellular communications. Some common examples are smart grids, GPS-based asset tracking systems, and smart lighting systems.
 - *Long life, battery powered applications:* For applications that need a long range. Some examples are water meters, gas detectors, smart agriculture systems, and so on.

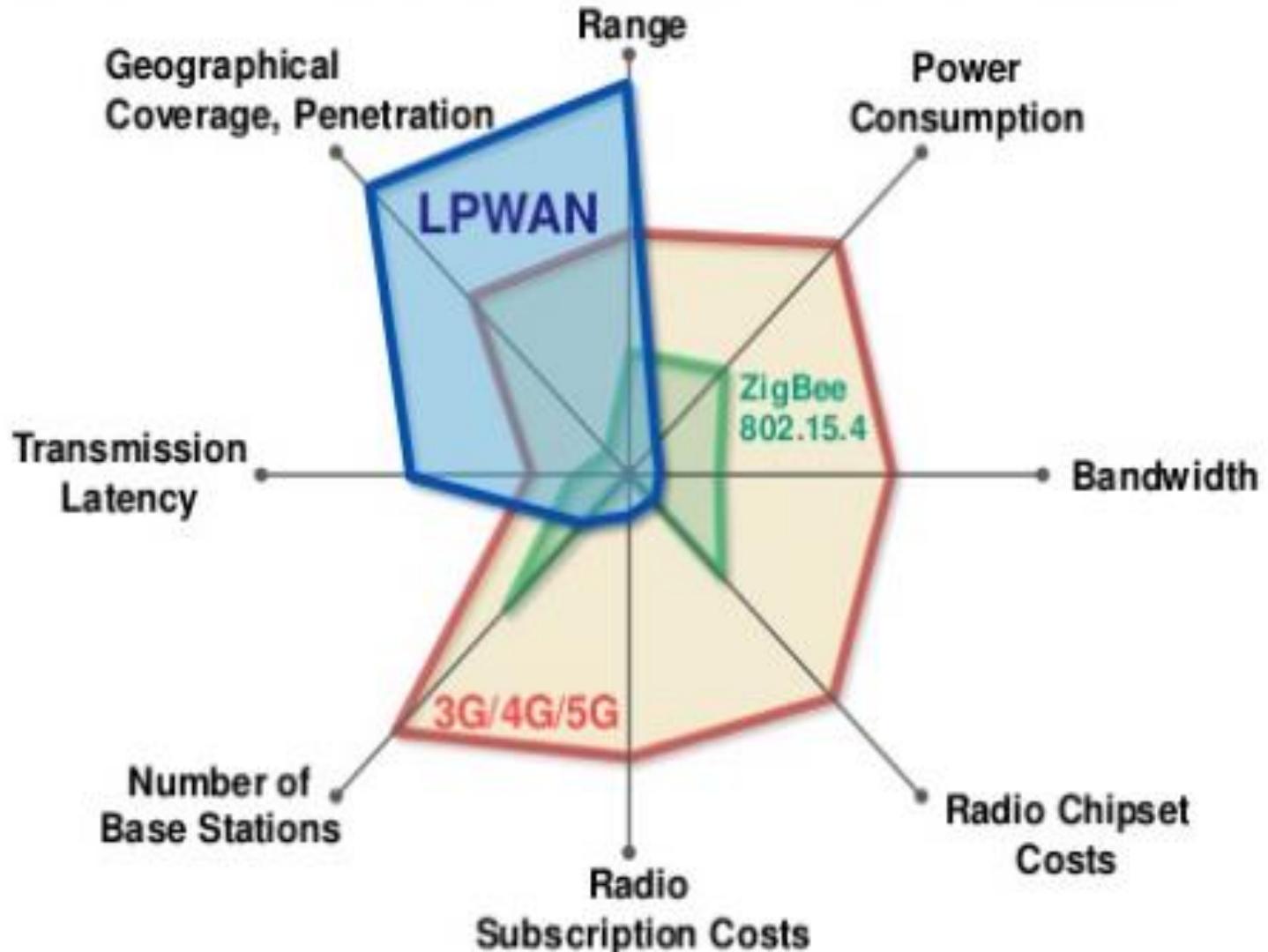
Bandwidth & Range Requirements of LPWAN Technologies

Bandwidth required



- The need of IoT and M2M applications have given rise to some specific requirements on LPWAN technologies as compared with other wireless technologies.

Requirements of LPWAN technology



LPWAN technology

- For LPWAN technology, a star-topology network is preferred instead of a mesh-topology network.
- The endpoints of star networks are connected directly to access points.
- Repeaters can be used to fill in gaps in coverage areas so that there is no drop in transmission power.
- LPWAN technologies operate with 140–160 decibels that can provide several miles of range.

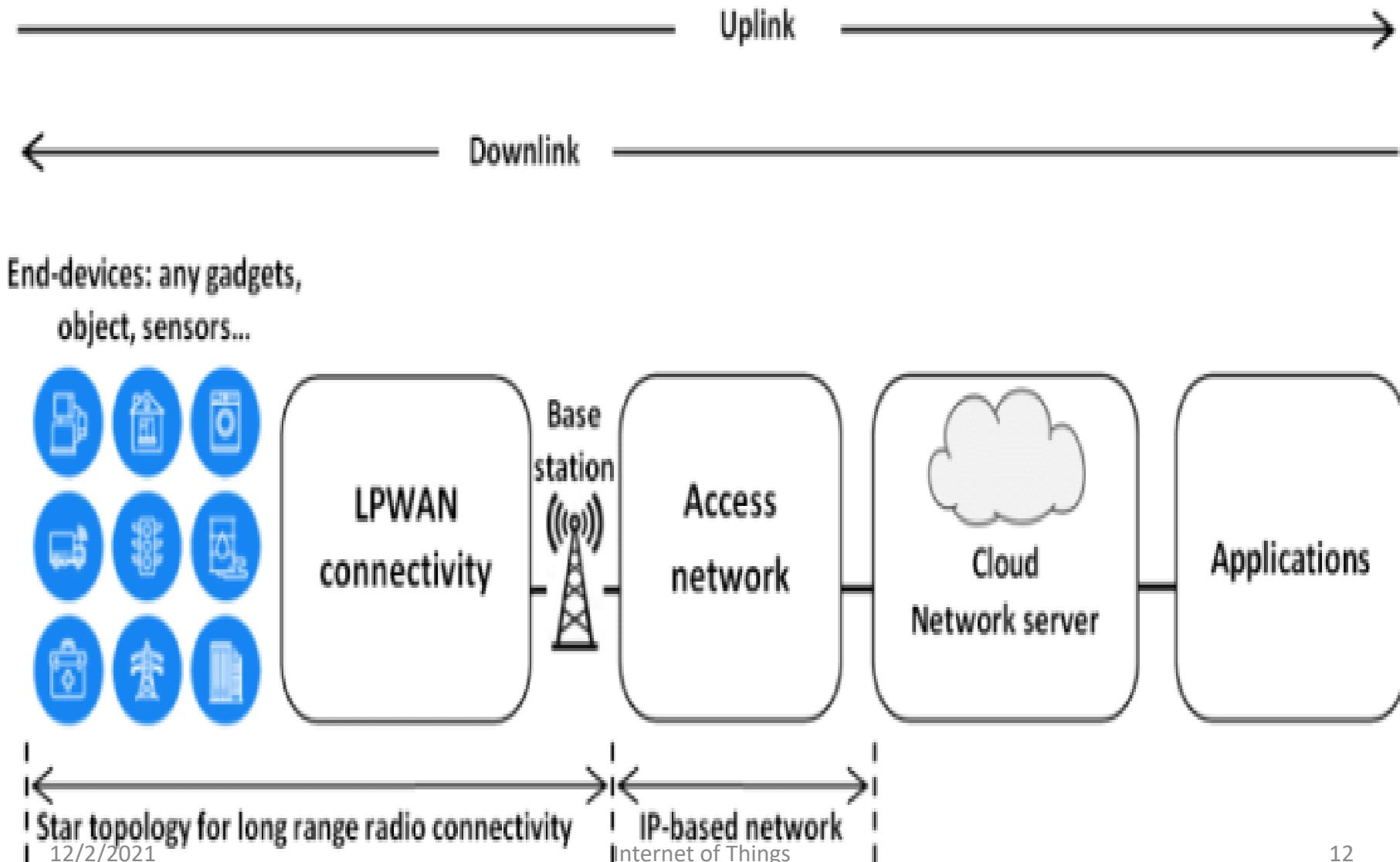
LPWAN technology

- In order to accomplish high miles, highly sensitive receivers are employed in LPWAN technologies.
- The following are the most important parameters to be considered while choosing a LPWAN technology:
 - Capacity
 - Quality of service
 - Range
 - Reliability
 - Battery life
 - Security
 - Cost
 - Proprietary versus standard

LPWAN Network Topologies

- LPWAN has two network topologies:
 - Direct device connectivity (base station)
 - Indirect device connectivity through an LPWAN gateway

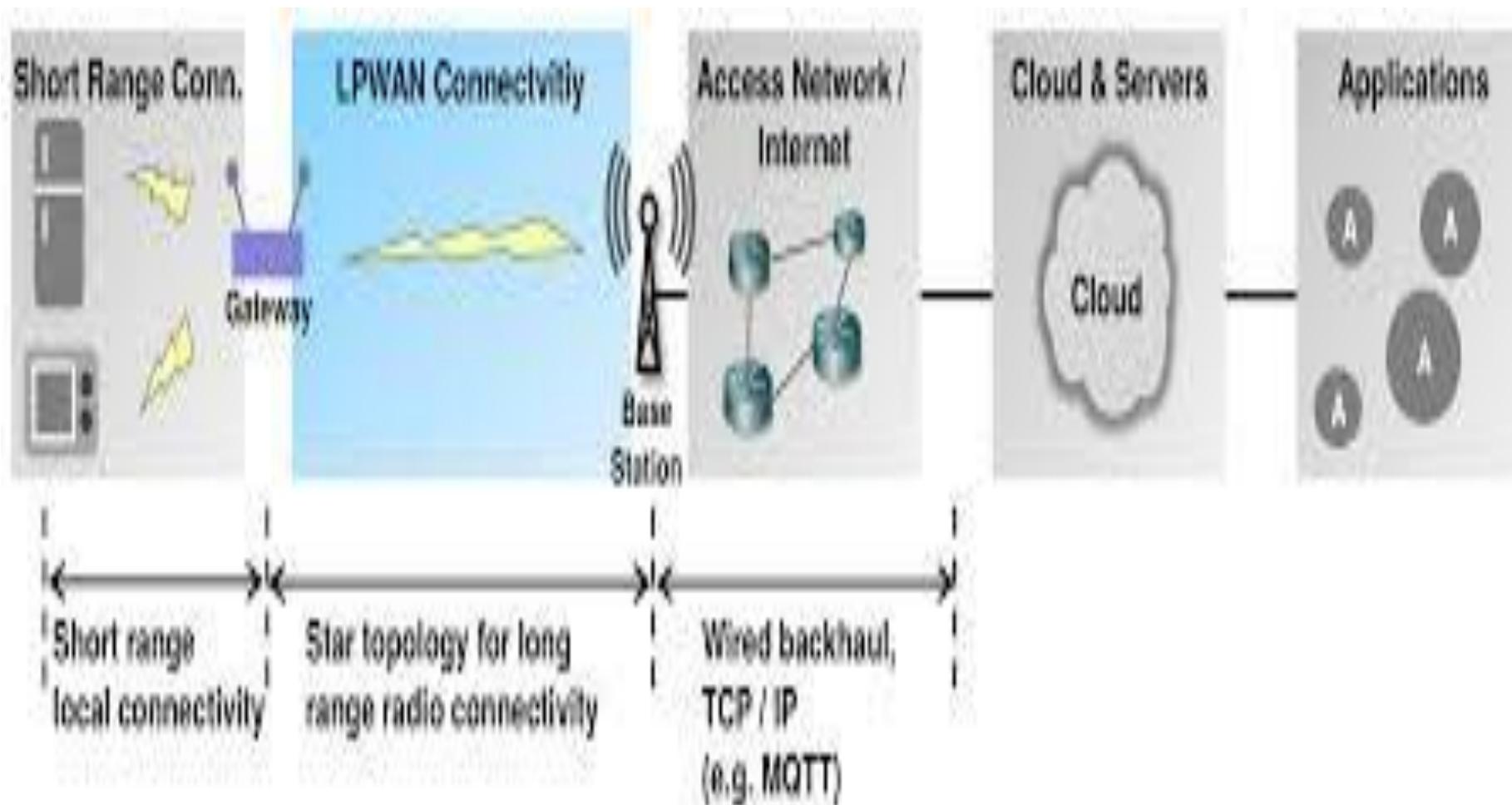
Direct device connectivity topology of LPWAN



Topology Description

- The base station that is present in the network provides connectivity to a large number of devices.
- The traffic is sent to servers (cloud) through TCP or IP-based networks (Internet).
- The base station is responsible for translation of protocol from IoT protocols such as MQTT or CoAP to specific device application protocols.

Indirect device connectivity through LPWAN gateway



Topology Description

- In certain networks where it is not possible to connect devices directly to LPWAN, a local gateway is used to bridge LPWAN connectivity to some short-range radio (SRD) technology like ZigBee or BLE.
- This gateway generally runs on mains power as it has to support a large number of devices.
- The gateway should also have the capability to perform protocol conversion from SRD radio technologies to LPWAN technology.
- Gateways also provide more security to IoT ecosystem as they offer options to implement powerful security algorithms.

Types of LPWANs

- LPWAN is not a single technology, but a group of various low-power, wide area network technologies. LPWANs can use licensed or unlicensed frequencies and include proprietary or open standard options.

SigFox

- The proprietary, unlicensed Sigfox is one of the most widely deployed LPWANs today. Running over a public network in the 868 MHz or 902 MHz bands, the ultra-narrowband technology only allows a single operator per country. While it can deliver messages over distances of 30-50 km in rural areas, 3-10 km in urban settings and up to 1,000 km in line-of-site applications, its packet size is limited to 150 messages of 12 bytes each per day. Downlink packets are smaller, limited to four messages of 8 bytes per day. Sending data back to endpoints can also be prone to interference. (Indian ocean connect)

RPMA

- Random phase multiple access, or RPMA, is a proprietary LPWAN from Ingenu Inc. Though it has a shorter range (up to 50 km line of sight and with 5-10 km nonline of sight), it offers better bidirectional communication than Sigfox. However, because it runs in the 2.4 GHz spectrum, it is prone to interference from Wi-Fi, Bluetooth and physical structures. It also typically has higher power consumption than other LPWAN options.

LoRa

- **LoRa** (Long Range) is a low-power wide-area network (LPWAN) protocol developed by Semtech.
- It is based on chirp spread spectrum (CSS) modulation techniques.
- LoRa Alliance¹² promotes use of an open standard for LoRa-based networks called LoRaWAN.
- LoRaWAN is the media access control (MAC) layer protocol that manages communication between LPWAN devices and gateways

LoRa

- LoRa allows users to define packet size.
- While open source, the underlying transceiver chip used to implement LoRa is only available from Semtech Corporation, the company behind the technology.
- Transmits in several sub-gigahertz frequencies, making it less prone to interference.

LoRa

- Following are the main features of LoRaWAN:
 - They have three open standards that provide various types of options for end users.
 - This standard lacks many features like support for roaming, packetization, firmware upgrades over air, and so on.
 - In order to use this standard, the network server software should be run in the cloud that mandates subscription from a network server vendor.

LoRa

- LoRa is ideal for applications that transmit small chunks of data with low bit rates. Data can be transmitted at a longer range compared to technologies like WiFi, Bluetooth or ZigBee. These features make LoRa well suited for sensors and actuators that operate in low power mode.

LoRa

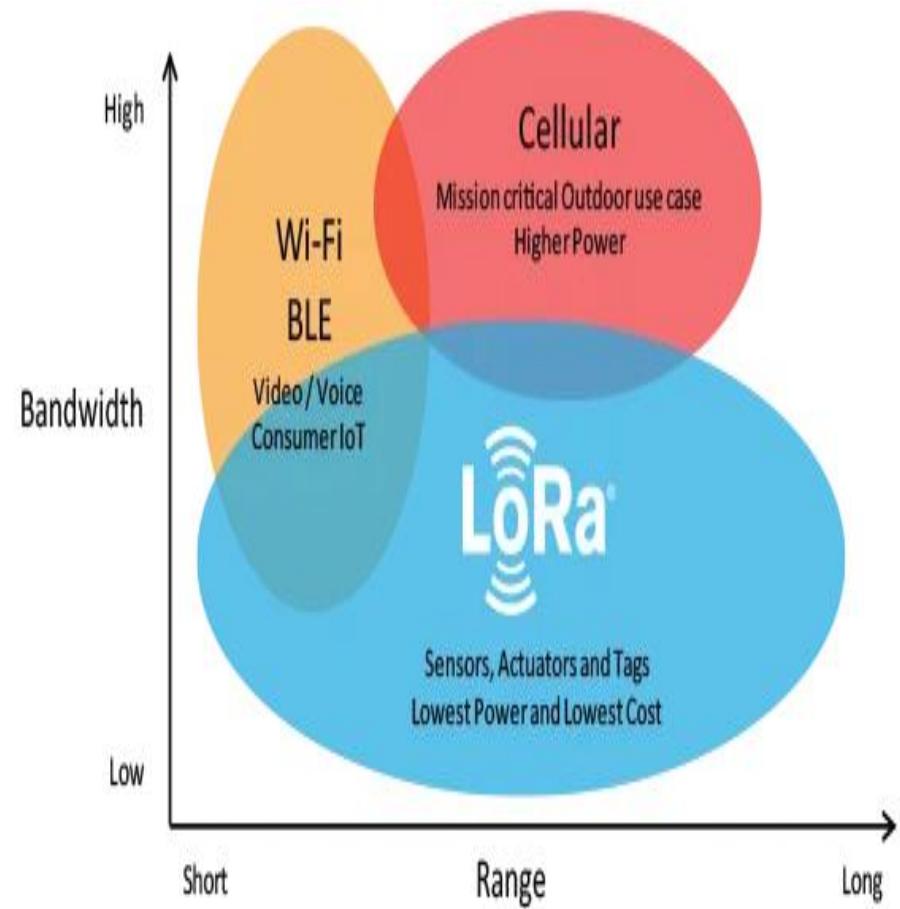
- LoRa can be operated on the license free **sub-gigahertz** bands, for example, 915 MHz, 868 MHz, and 433 MHz. It also can be operated on **2.4 GHz** to achieve higher data rates compared to sub-gigahertz bands, at the cost of range. These frequencies fall into ISM bands that are reserved internationally for industrial, scientific, and medical purposes.

LoRaWAN

- LoRaWAN is a Media Access Control (MAC) layer protocol built on top of LoRa modulation. It is a software layer which defines how devices use the LoRa hardware, for example when they transmit, and the format of messages.

Bandwidth vs Range

- LoRaWAN is suitable for transmitting small size payloads (like sensor data) over long distances. LoRa modulation provides a significantly greater communication range with low bandwidths than other competing wireless data transmission technologies.



Why LoRaWAN

- **Ultra low power** - LoRaWAN end devices are optimized to operate in low power mode and can last up to 10 years on a single coin cell battery.
- **Long range** - LoRaWAN gateways can transmit and receive signals over a distance of over 10 kilometers in rural areas and up to 3 kilometers in dense urban areas.
- **Deep indoor penetration** - LoRaWAN networks can provide deep indoor coverage, and easily cover multi floor buildings.
- **License free spectrum** - You don't have to pay expensive frequency spectrum license fees to deploy a LoRaWAN network.
- **Geolocation**- A LoRaWAN network can determine the location of end devices using triangulation without the need for GPS. A LoRa end device can be located if at least three gateways pick up its signal.
- **High capacity** - LoRaWAN Network Servers handle millions of messages from thousands of gateways.

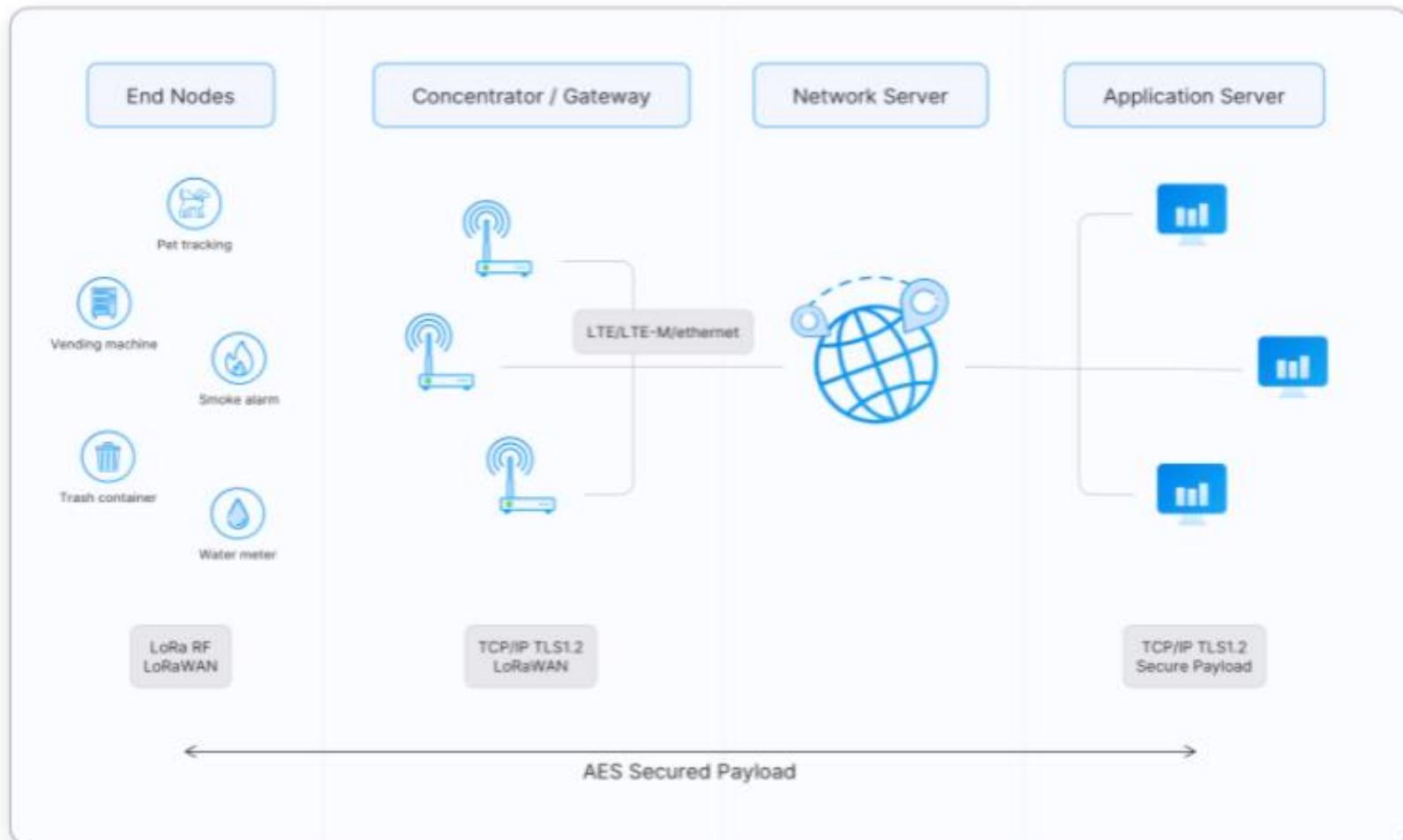
Why LoRaWAN

- **End-to-end security**- LoRaWAN ensures secure communication between the end device and the application server using AES-128 encryption.
- **Roaming**- LoRaWAN end devices can perform seamless handovers from one network to another.
- **Low cost** - Minimal infrastructure, low-cost end nodes and open source software.
- **Ecosystem**- LoRaWAN has a very large ecosystem of device makers, gateway makers, antenna makers, network service providers, and application developers

LoRaWAN Architecture

- LoRaWAN is a media access control (MAC) protocol for wide area networks.
- It is designed to allow low-powered devices to communicate with Internet-connected applications over long range wireless connections.
- LoRaWAN can be mapped to the second and third layer of the OSI model.
- It is implemented on top of LoRa or FSK modulation in industrial, scientific and medical (ISM) radio bands.

LoRaWAN Architecture



LoRaWAN Architecture

- **End Device, Node, Mote** - an object with an embedded low-power communication device.
- **Gateway** - antennas that receive broadcasts from End Devices and send data back to End Devices.
- **Network Server** - servers that route messages from End Devices to the right Application, and back.
- **Application** - a piece of software, running on a server.

LoRaWAN Architecture

- End devices communicate with nearby gateways and each gateway is connected to the network server. LoRaWAN networks use an ALOHA based protocol, so end devices don't need to peer with specific gateways. Messages sent from end devices travel through all gateways within range. Message deduplication is handled by the network server.

End devices

- A LoRaWAN end device can be a sensor, an actuator, or both.
- They are often battery operated.
- These end devices are wirelessly connected to the LoRaWAN network through gateways using LoRa RF modulation.

Example end device - a water meter.



Gateways

- Each gateway is registered to a LoRaWAN network.
- A gateway receives LoRa messages from end devices and simply forwards them to the LoRaWAN network server.
- Gateways are connected to the internet through an IP backbone.
- IP traffic from gateway to the network server can be **backhauled** through Cellular (3G/4G/5G), WiFi, Ethernet, Fiber-optic or 2.4 GHz radio links

Types of LoRaWAN Gateways

- LoRaWAN gateways can be categorized into indoor (picocell) and outdoor (macrocell) gateways. Indoor gateways provide coverage in difficult-to-reach indoor locations and are therefore suitable for use in homes, businesses and buildings.

Example Gateway



LoRaWAN Gateway

- Outdoor gateways are suitable for providing coverage in rural, urban, and dense urban areas. This type of gateway is intended for deployment places like cellular towers, the rooftops of very tall buildings, metal pipes (masts) etc.
- Usually, the receiver sensitivity of an outdoor gateway is higher than the receiver sensitivity of an indoor gateway

LoRaWAN outdoor gateway



Network Server

- The network server manages the entire LoRaWAN network. It receives IP traffic from gateways. The Network Server is responsible for network management functions like:
- **Over-The-Air-Activation**
- **Message deduplication**
- **Message routing**
- **Adaptive data rate control**
- **Acknowledgement of messages**

Application Server

- The application server processes application-specific data messages received from end devices.
- It also generates all the application-layer downlink payloads and sends them to the connected end devices through the network server.
- A LoRaWAN network can have more than one application server. The collected data can be interpreted by applying techniques like machine learning and artificial intelligence to solve business problems.

Join Server

- The Join Server processes join-request messages sent by end devices.
- It stores root keys, generates session keys, and transfers those session keys to the network server and the application server.