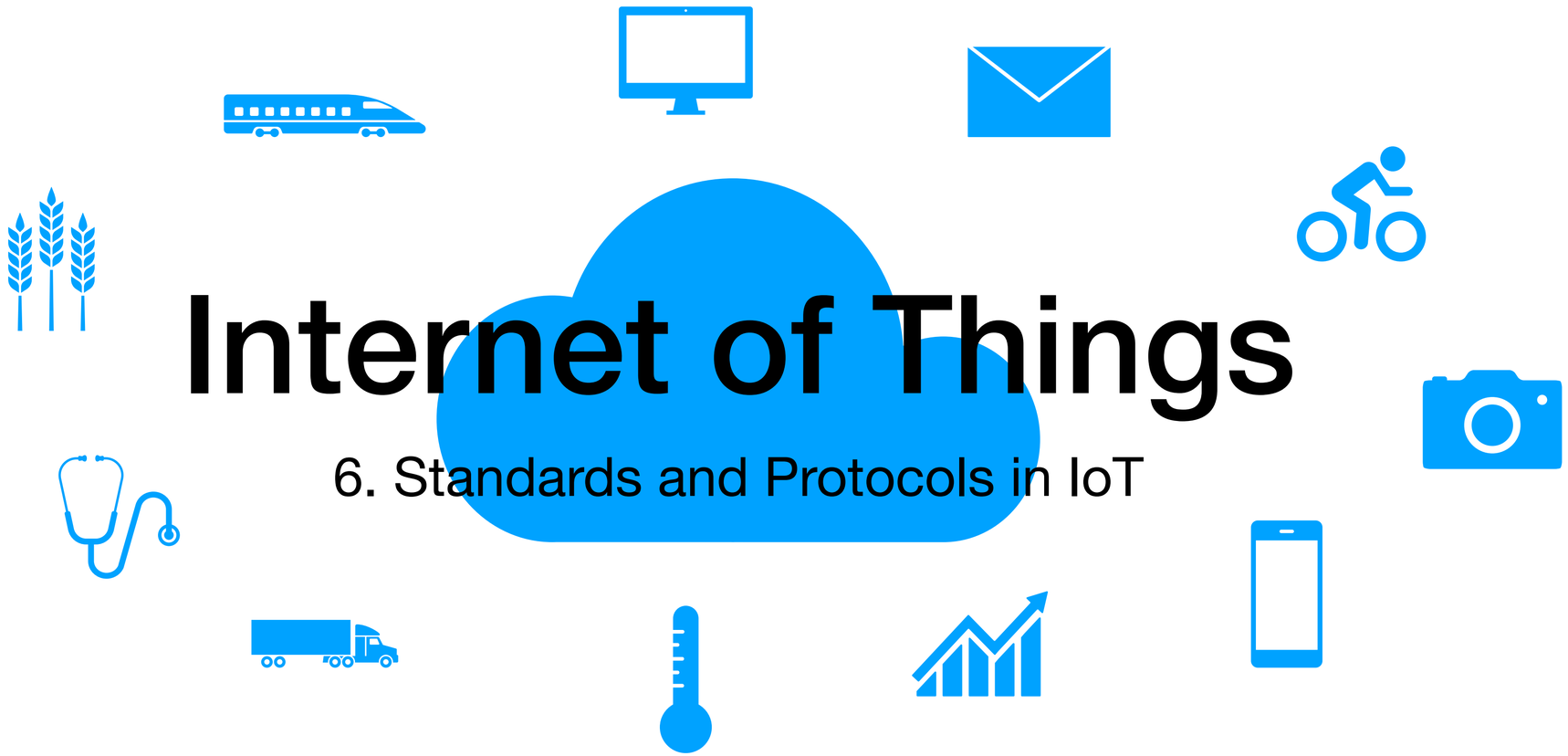


Internet of Things

6. Standards and Protocols in IoT

M.Sc. Computer Science 2024-2025

Viviana Arrigoni

6.2 Layer 3 Protocols

Advantages of IP in IoT

- The Internet protocol (IP) has been around for over 35 years!

Uniquitous

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

The Internet hosts millions of private and public IP infrastructures all around the world

Scalable

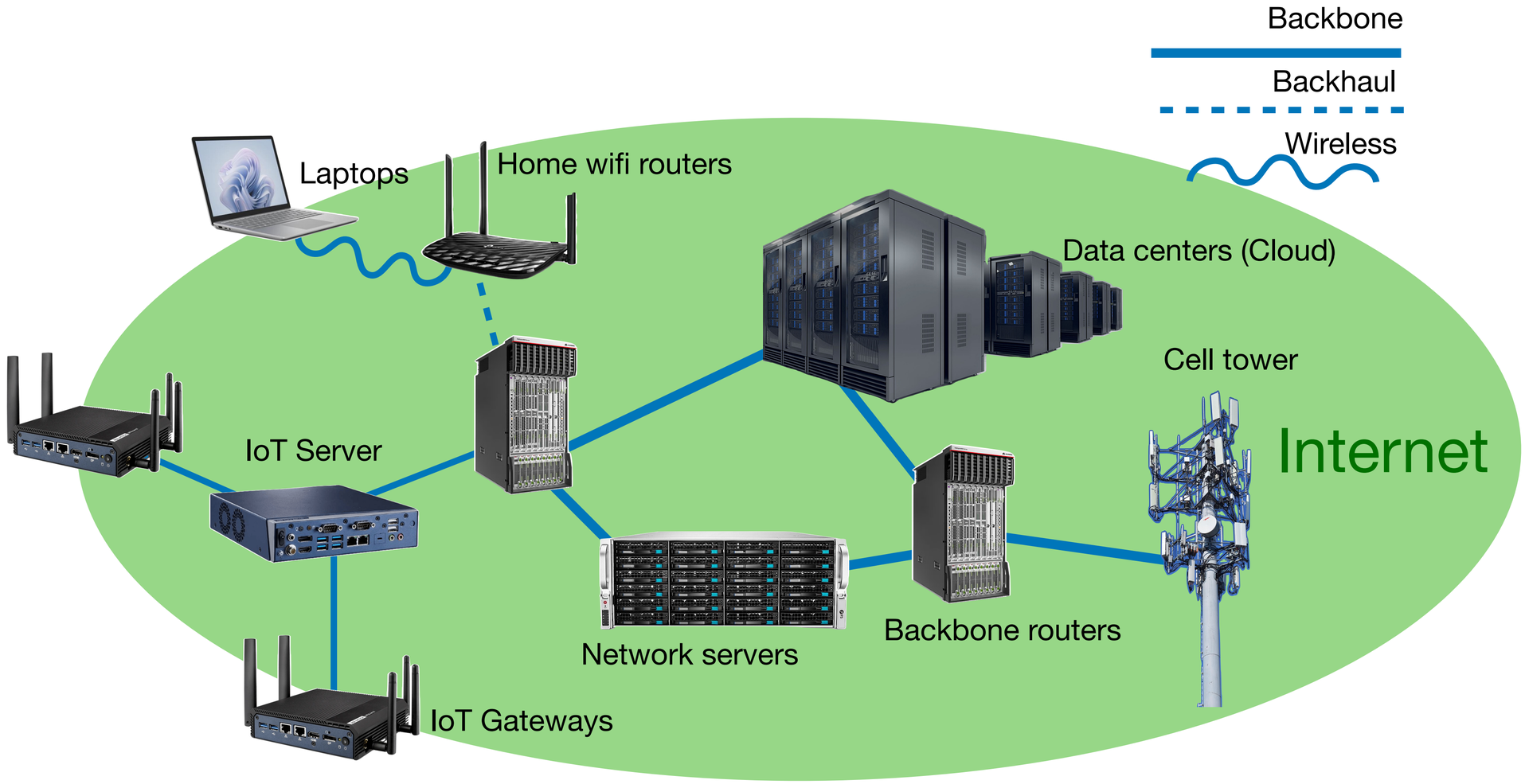
Manageable and highly secure

Network management and security protocols, mechanisms, and toolsets that are widely available.

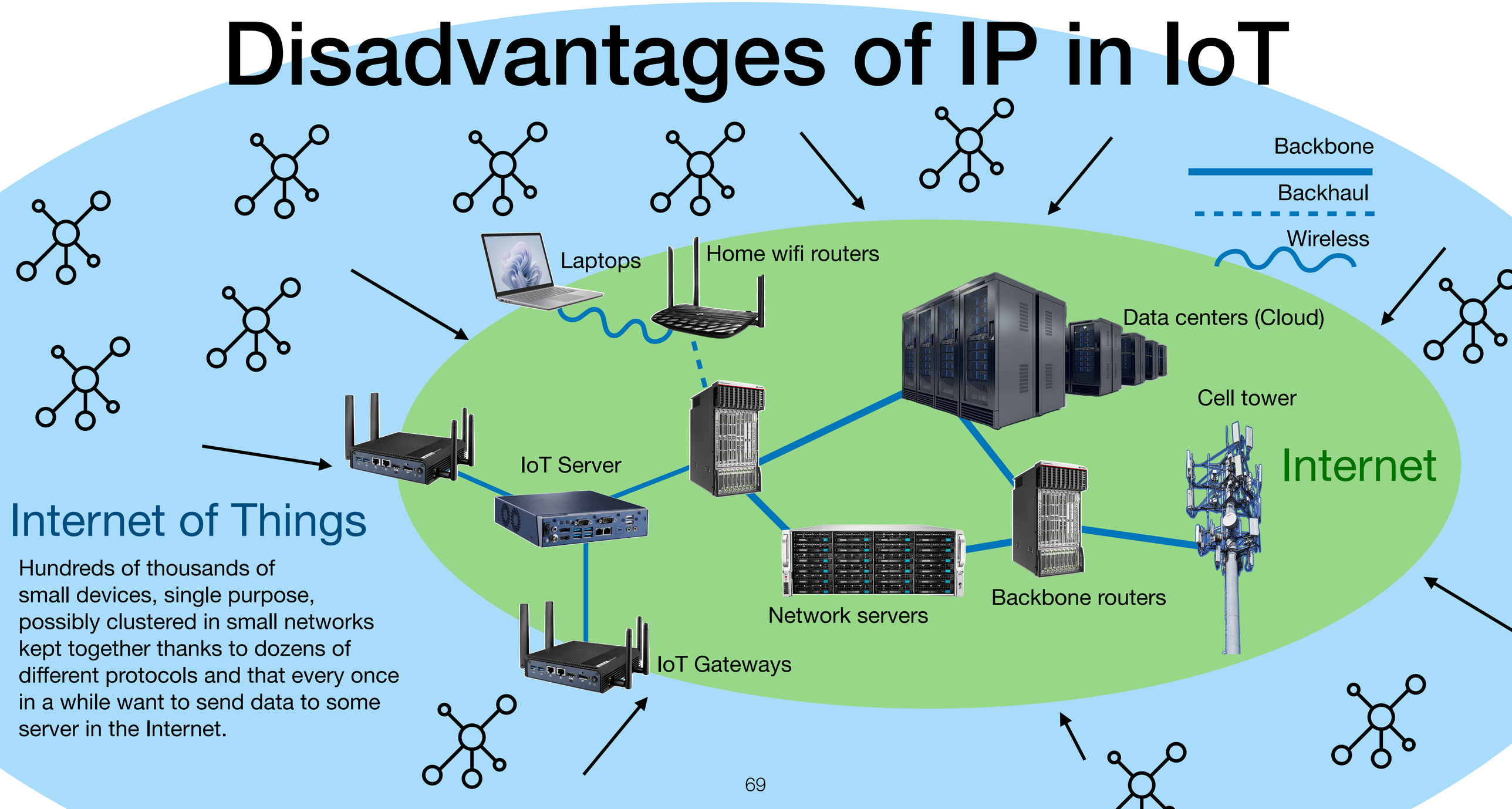
It has been used for years in critical infrastructures, such as financial and defence networks. It supports critical services and delay-sensitive applications (e.g., video calls, online gaming)

Stable and resilient

Disadvantages of IP in IoT



Disadvantages of IP in IoT



Internet of Things

Hundreds of thousands of small devices, single purpose, possibly clustered in small networks kept together thanks to dozens of different protocols and that every once in a while want to send data to some server in the Internet.

Disadvantages of IP in IoT

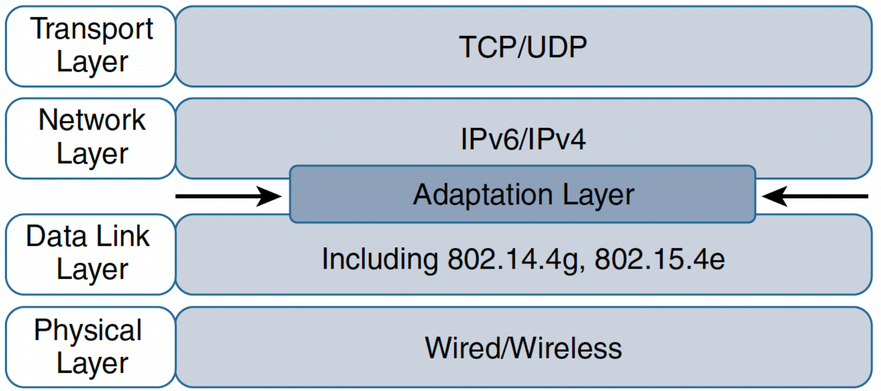
- Last mile communication (i.e., communication between sensors, embedded systems, IoT gateway, etc) has very different features than Internet communication.
  - **Bidirectional versus unidirectional data flow.** Some devices essentially only send data and reports.
  - **Overhead for last-mile communication.**
    IPv4 header ≥ 20 bytes
    IPv6 header = 40 bytes
    UDP header = 8 bytes
    TCP header ≥ 20 bytes.
    If the data to be forwarded by a device is infrequent and only a few bytes long, you can potentially have more header overhead than device data
  - **Network diversity.** Some devices support vendor-specific protocols and are not compatible with other protocols/devices.

IoT Devices

- IoT comprises devices that can be:
- Very constrained in resources:
  - may communicate infrequently to transmit a few bytes.
  - may have limited security and management capabilities.
- Similar to generic PCs in terms of computing and power resources but have constrained networking capacities, e.g., limited bandwidth.
  - These nodes usually implement a full IP stack, but network design must cope with the bandwidth constraints.

Optimising IP for IoT

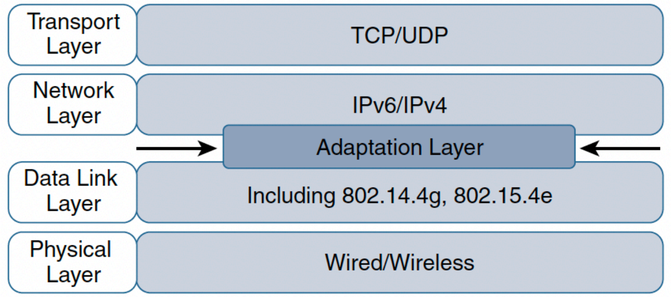
- IoT data travels through the Internet for:
  - Remote monitoring and controlling (e.g., data is sent to mobile phones through cellular communication).
  - Cloud analytics and storage.
  - Integration with external services.
  - Firmware updates/configuration.
- The Internet Protocol is key for a successful Internet of Things, but constrained nodes and networks mandate optimisation at various layers and on multiple protocols of the IP architecture.
- Optimisation can be achieved by adding an **adaptation layer** in the TCP/IP stack



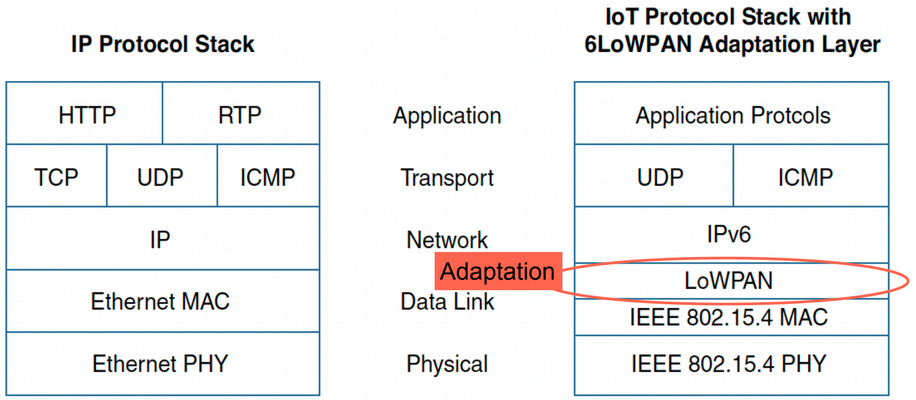
6.2 Layer 3 Protocols
6.2.1 6LoWPAN

6LoWPAN (1)

- 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) is an open standard defined in RFC 6282 by the Internet Engineering Task Force (IETF), originally conceived to support IEEE 802.15.4 low-power wireless networks in the 2.4-GHz band.
  - It is now being adapted and used over a variety of other networking media including Sub-GHz low-power RF.

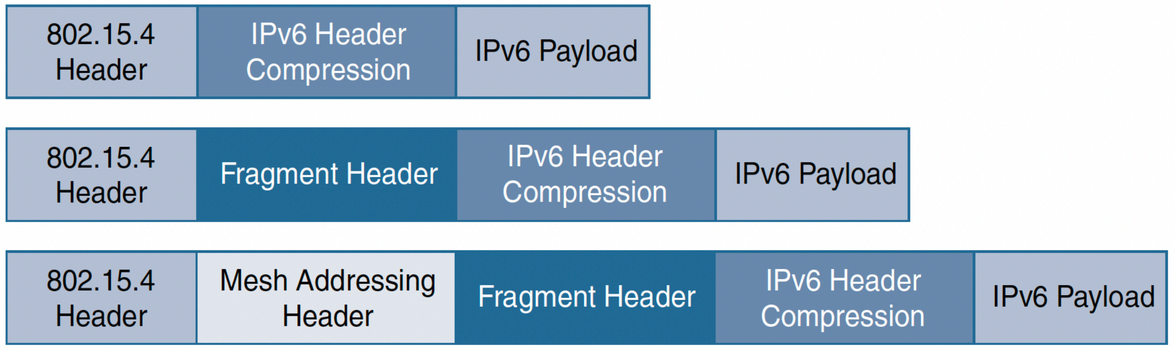


- It allows transmission of IP packets in low-power wireless networks.
- Works at the **adaptation layer**



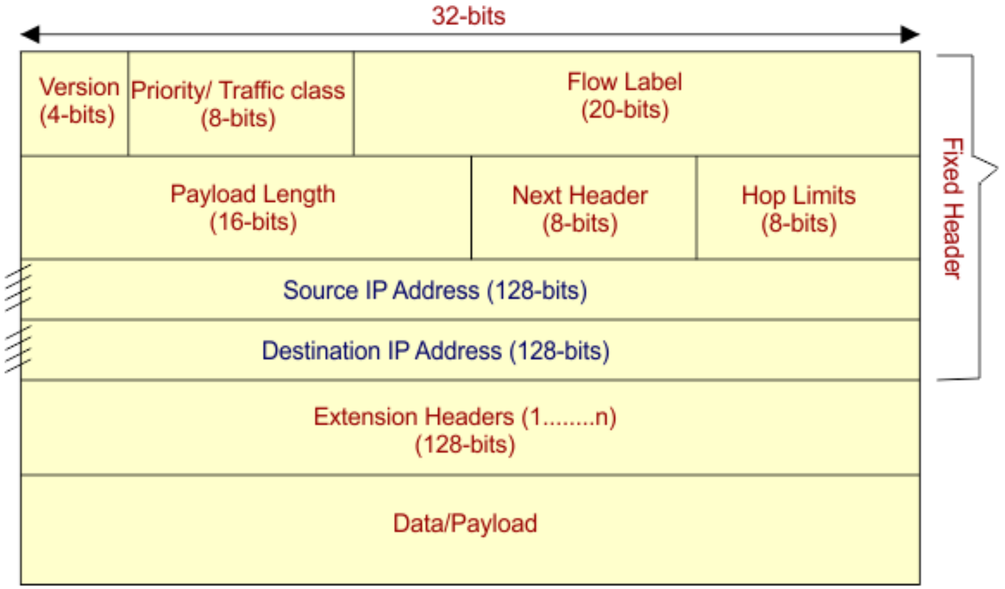
6LoWPAN (2)

- The optimisation and adaptation carried out with 6LoWPAN is achieved thanks to **header compression**, **fragmentation**, and **mesh addressing**.
- Such operations create separated headers stacked in the adaptation layer.
- Depending on the implementation, all, none, or any combination of these capabilities and their corresponding headers can be enabled.
- 6LoWPAN only works for **IPv6**, does not support IPV4.



IPv6

- IPv6 is the latest version of the Internet Protocol and is designed to replace IPv4, offering a much larger address space.
- IPv6 header is 40 byte long.
- **Version.** Always 6 = 0110.
- **Traffic Class.** Indicates class or priority of the IPv6 packet.
- **Flow Label.** Used by a source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers. “0” is used for no specific flow.



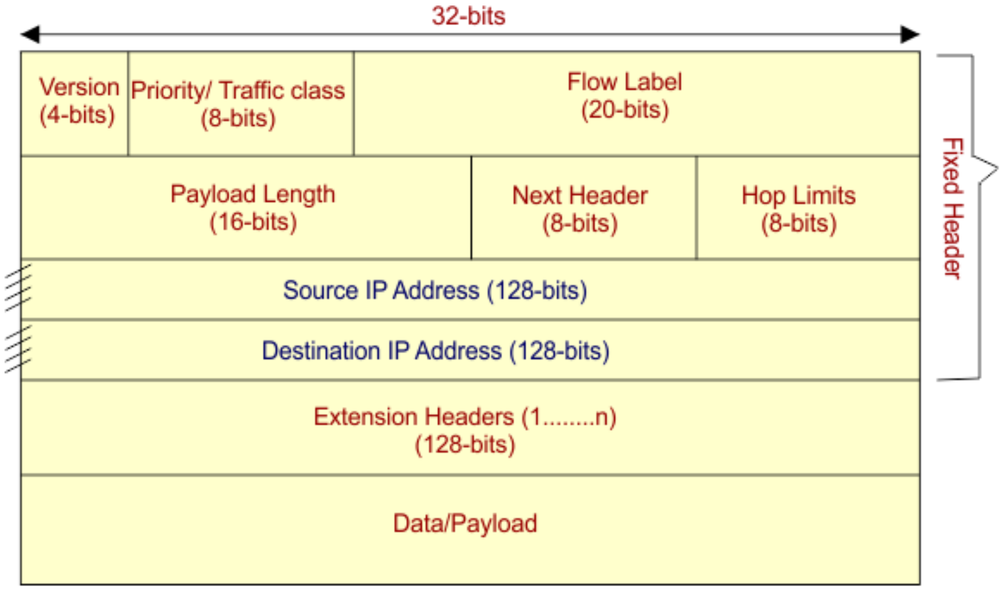
IPV6 Header

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

A flow is a stream of packets that need consistent treatment (e.g., video stream). A flow is characterised by source and destination, protocol, port. Using traffic IDs allows to make traffic engineering more efficient.

IPv6

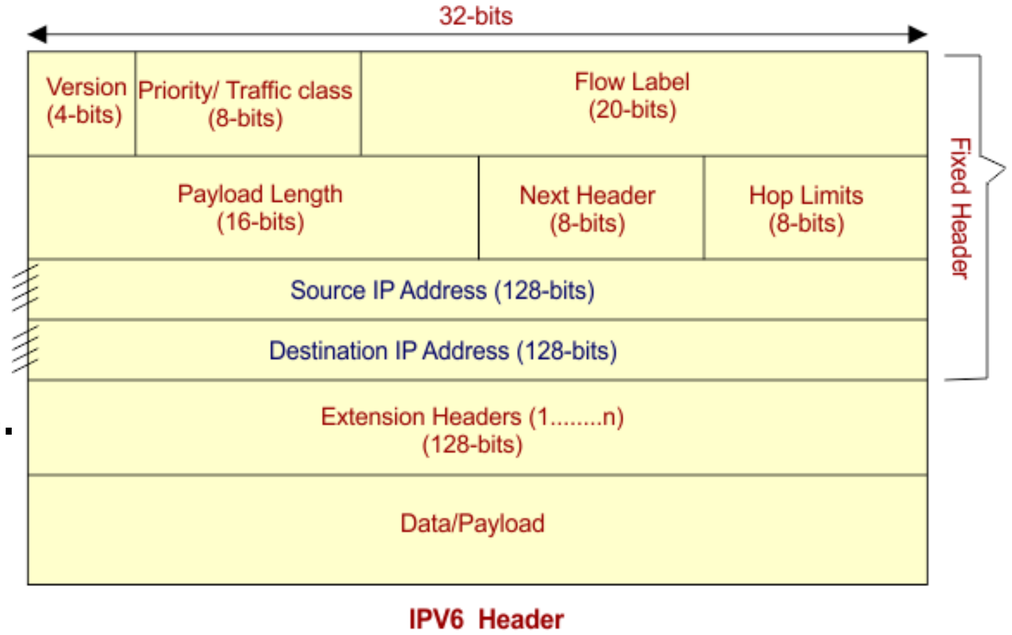
- IPv6 is the latest version of the Internet Protocol and is designed to replace IPv4, offering a much larger address space.
- IPv6 header is 40 byte long.
- **Version.** Always 6 = 0110.
- **Traffic Class.** Indicates class or priority of the IPv6 packet.
- **Flow Label.** Used by a source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers. “0” is used for no specific flow.
- **Payload Length.** Indicates the total size of the payload



IPV6 Header

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

IPv6



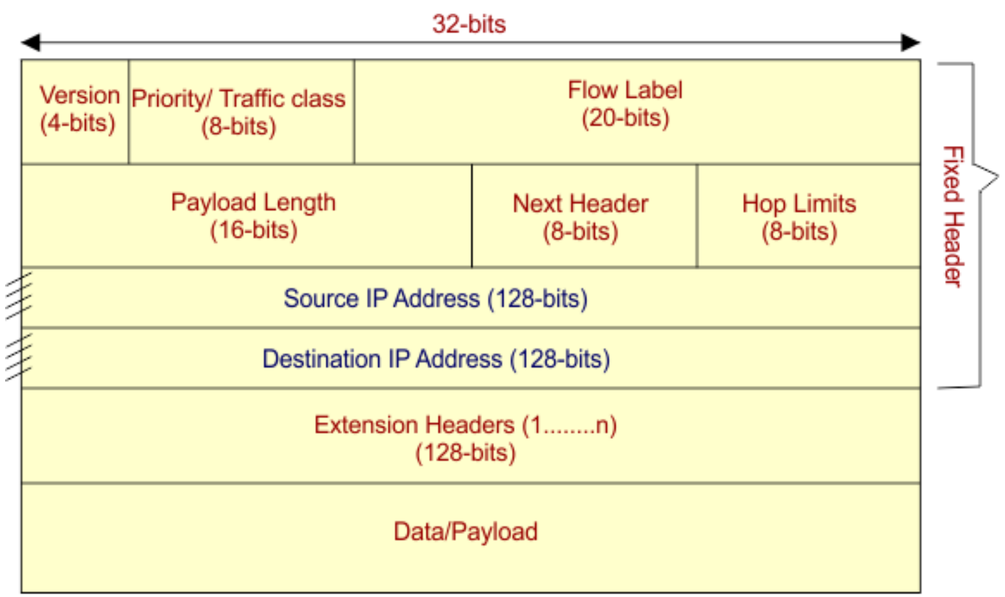
IPV6 Header

- **Next header.** Usually specifies the transport layer protocol in the datagram's payload (UDP/TCP). When extension headers (EH) are present in the packet, this field indicates which EH follows.
- **Hop Limits.** Replaces TTL in IPv4.
- **Source and Destination IPv6 addresses.**
- **Extension headers.** Located between the IPv6 header and the transport layer header and contain IPv6 options.
  - Many IPv6 extension headers are not examined or processed by any router along a packet's delivery path until the packet arrives at its final destination (more efficient/faster forwarding).
  - Include fragmentation, authentication, destination, hop-by-hop options, and others.

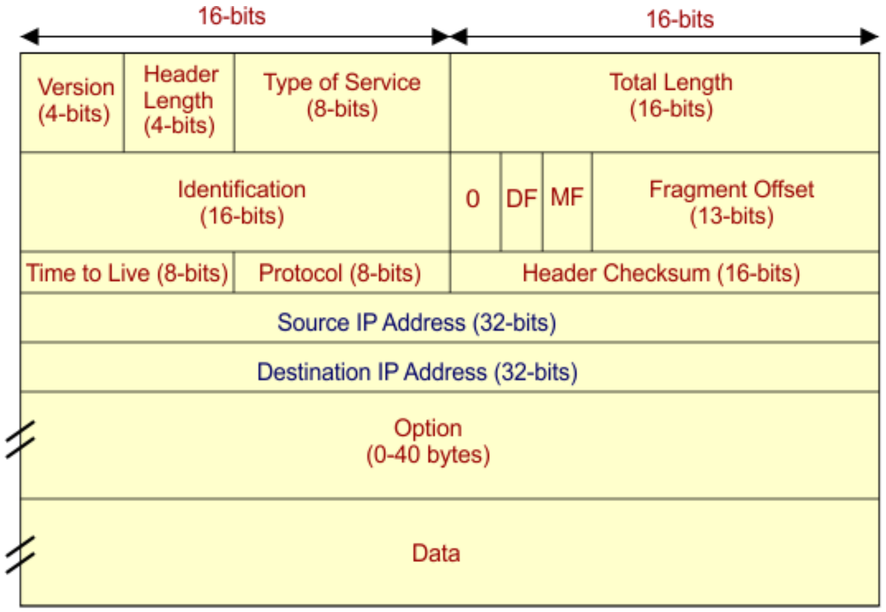
IPv6

- **IPv6 packet header removes the following entries from IPv4:**
- **Header checksum**, which is replicated in MAC header, no needed in IP.
- **Fragment offset.** IPv4 routers can fragment a data packet they have to forward if the packet is larger than the next hop's Maximum Transmission Unit (MTU). The fragments are reassembled by the receiving host.
  - In IPv6, **routers do not support fragmentation**, which is carried out only by end devices that determine the MTU through Path MTU Discovery before sending packets.
    - Path MTU Discovery is a standardized technique for determining the maximum transmission unit (MTU) size of the path between two IP hosts.
- **Options** in IPv4 are replaced by pointer to next header extension.

IPv6 vs IPv4



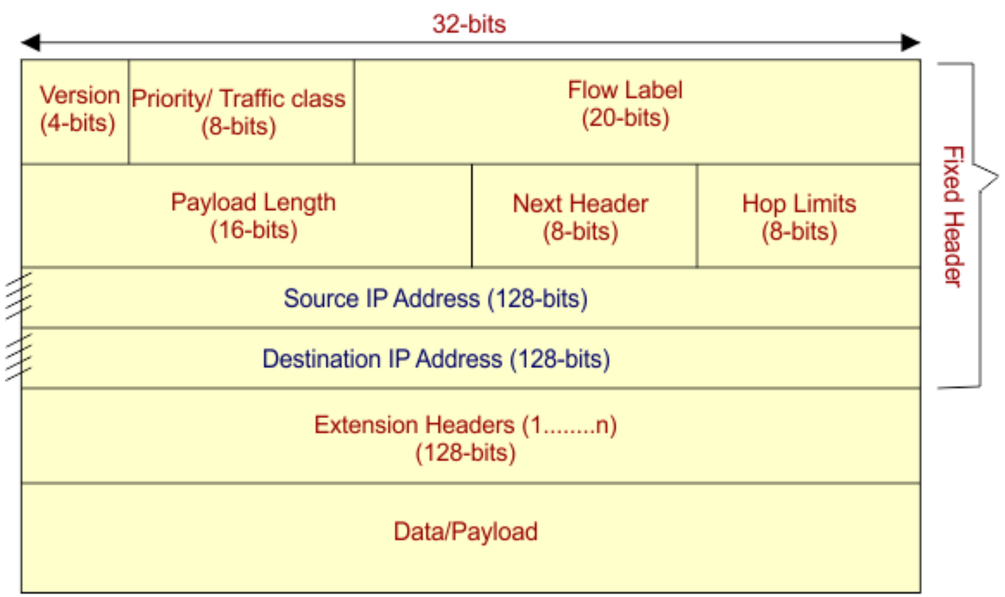
IPV6 Header



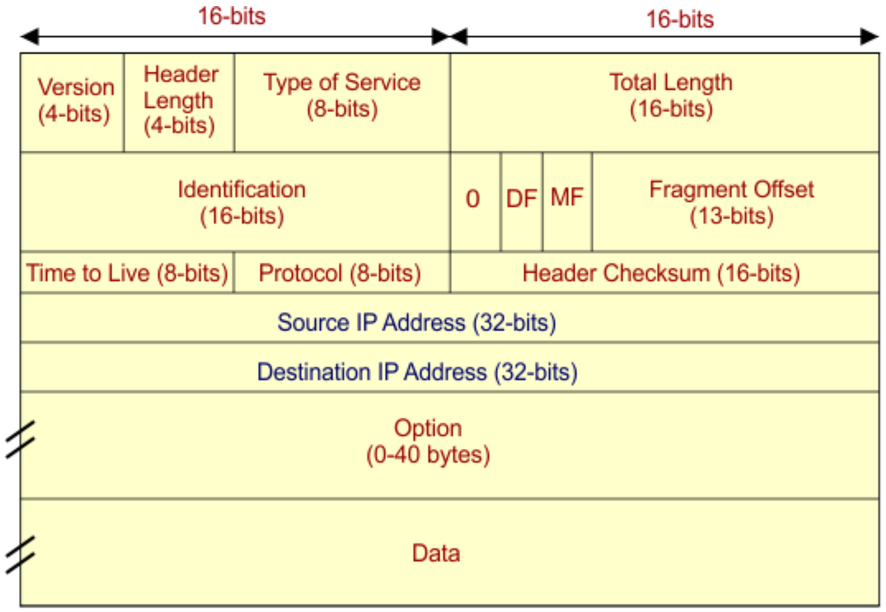
IPV4 Datagram Header

- In IPv4 the header length is variable, in IPv6 it is always 40 bytes.
- TOS in IPv4 is used for quality of service; in IPv6, this is replaced by Traffic Class and Flow Label.
- In IPv4, Total Length indicates the size of the packet. Replaced in IPv6 by Payload Length field.
- Identification+flags+fragment offset in IPv4 are for fragmentation.

IPv6 vs IPv4



IPV6 Header



IPV4 Datagram Header

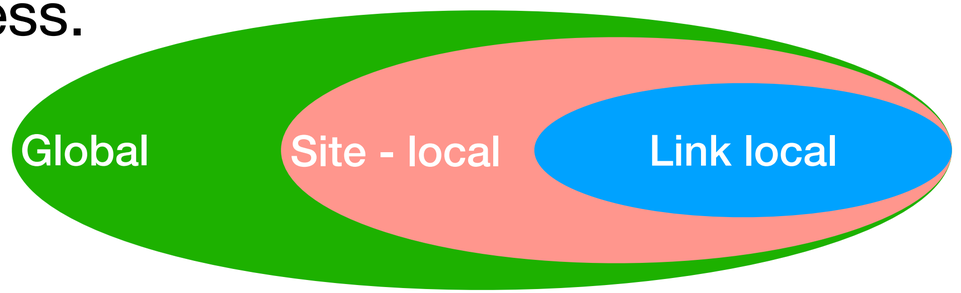
- TTL is replaced by hop limits.
- In IPV4, Protocol indicates the higher-layer protocol (TCP, UDP). In IPv6, this is specified by the next header field. The next header can be a higher-layer protocol or an extension header.
- Header Checksum is omitted in IPv6
- IPv4 allows optional fields, but in IPv6, options are handled by extension headers.

IPv6 Addressing (1)

- IPv6 addresses are **128 bit long**, arranged in eight groups written in hexadecimal, each of which is 16 bits (IPv4 addresses are 32 bit long).
  example: 3FFE:085B:1F1F:0000:0000:0000:00A9:1234
- There are three IPv6 address types:
  - **Unicast** (one to one communication between two specific endpoints).
  - **Anycast** (a single IP address is shared by devices in multiple locations. Routers direct packets addressed to the destination whose location is nearest the sender).
    - Examples: DNS servers, Content delivery networks.
  - **Multicast** (data transmission is addressed to a group of destination computers simultaneously).

IPv6 Addressing (2)

- An IPv6 network interface has multiple addresses:
- **Link local address:**
  - Contains the interface identifier (computed from the MAC address).
  - Can be used to reach the neighbouring nodes attached to the same link (1-hop distant, mostly used for neighbour discovery, router discovery, and local communication).
- **Site-local address:**
  - Has similar properties as an IPv4 private address.
- **Global address:**
  - Used to route IP datagrams over the Internet.



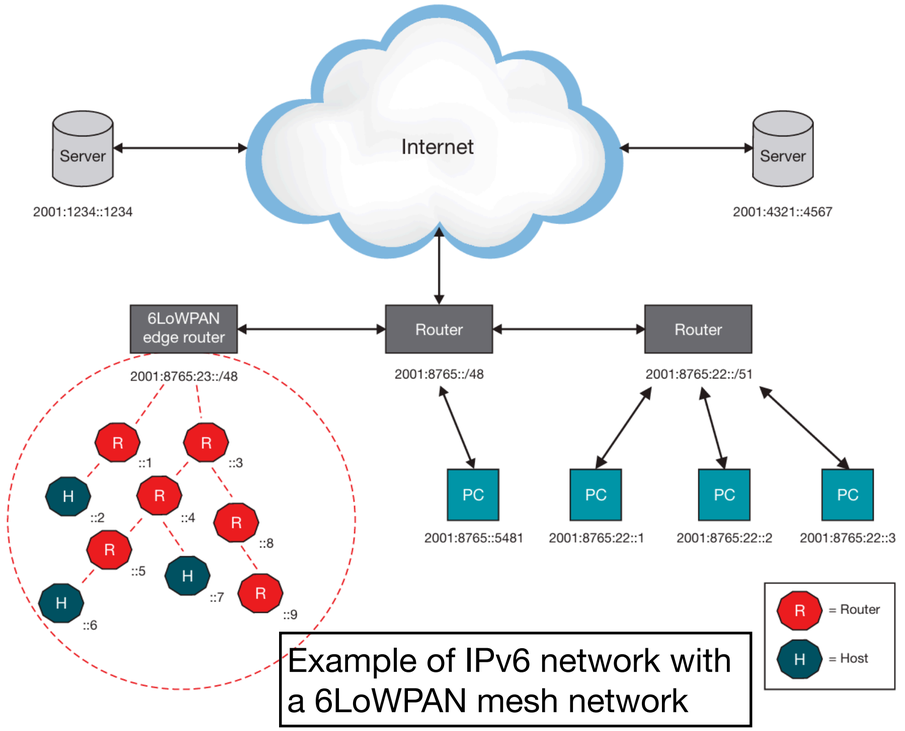
IPv6 Addressing (3)

- Why do we need a link-local address?
- IPv6 replaced ARP* (Address Resolution Protocol) with the Neighbour Discovery Protocol (NDP), which works over ICMPv6 (L3), not Ethernet (L2).
- Recall ICMP (Internet Control Message Protocol) in IPv4:
  - Delivers error messages (e.g., destination unreachable, time exceeded) and supports diagnostics (e.g., ping, traceroute).
- ICMPv6 supports many more functionalities beyond these (neighbour discovery, router discovery, prefix discovery, redirects, duplicate address detection etc).
- NDP works at a L3 layer and resolve a MAC address in IPv6 by sending ICMPv6 Neighbour Solicitation messages, which use link-local addresses.

* Recall that ARP is a link layer protocol used for discovering and mapping the link layer address (MAC address) associated with an IP address.

6LoWPAN networks

- The 6LoWPAN network is connected to the IPv6 network using an edge router.
- The edge router handles:
  1) Data exchange between 6LoWPAN devices and the Internet (or other IPv6 network).
  2) Local data exchange between devices inside the 6LoWPAN.
  3) Generation and maintenance of the 6LoWPAN network.



Example of IPv6 network with a 6LoWPAN mesh network

- By communicating natively with IP, 6LoWPAN networks are connected to other networks simply using IP routers.
- 6LoWPAN networks will typically operate on the edge, acting as **stub** networks (data going into the network is destined for one of the devices in the 6LoWPAN).

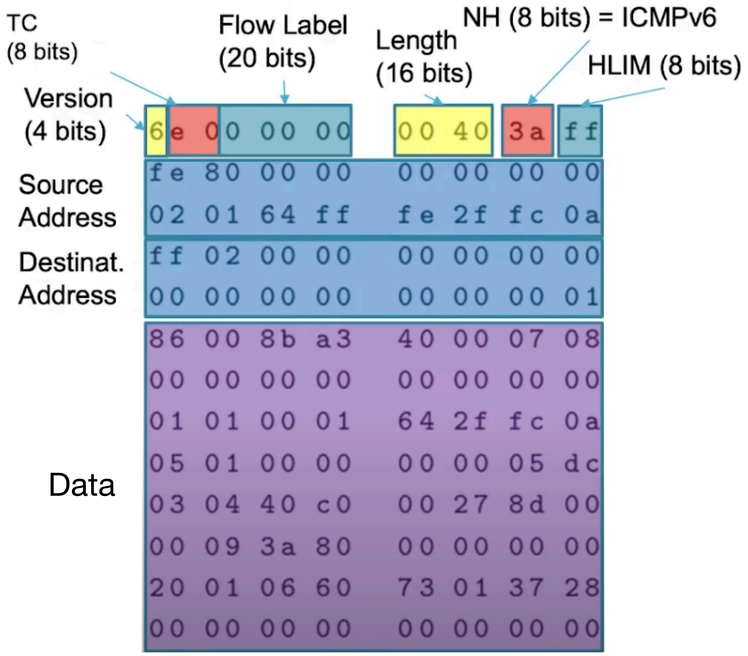
Adaptation Layer

- Goal: optimise the transmission of IPv6 packets over low-power and lossy networks (LLNs) such as IEEE 802.15.4.
- Worst case scenario:
  - Maximum size in IEEE 802.15.4 PSDU -> 127 bytes, which may carry:
    - Standard IPv6 header (40 bytes)
    - UDP header (8 bytes)
  - + all MAC frame’s headers and footers (around 25 bytes)
  - Only 54 bytes left for payload! -> That’s why we need adaptation!

Header Compression

- The way the headers can be compressed is one of the factors that led to 6LowPan only supporting IPv6 and not IPv4.
- 6LoWPAN header compression changes depending on the protocol release. At a high level:
  - 6LoWPAN compression works by taking advantage of shared information known by all nodes from their participation in the local network (contexts, prefixes).
  - It omits some standard header fields by assuming commonly used values.
  - Omits information that can be inferred from MAC headers.

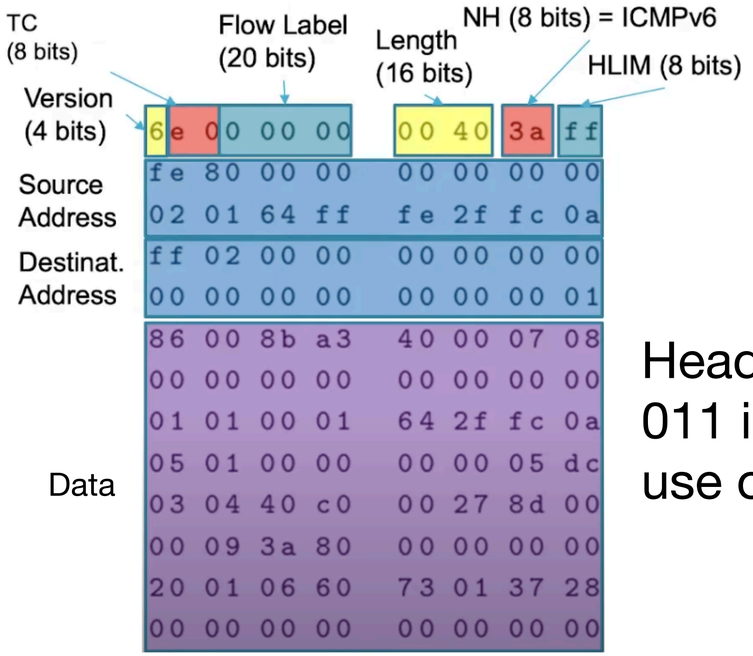
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Header Compression-scenario 1

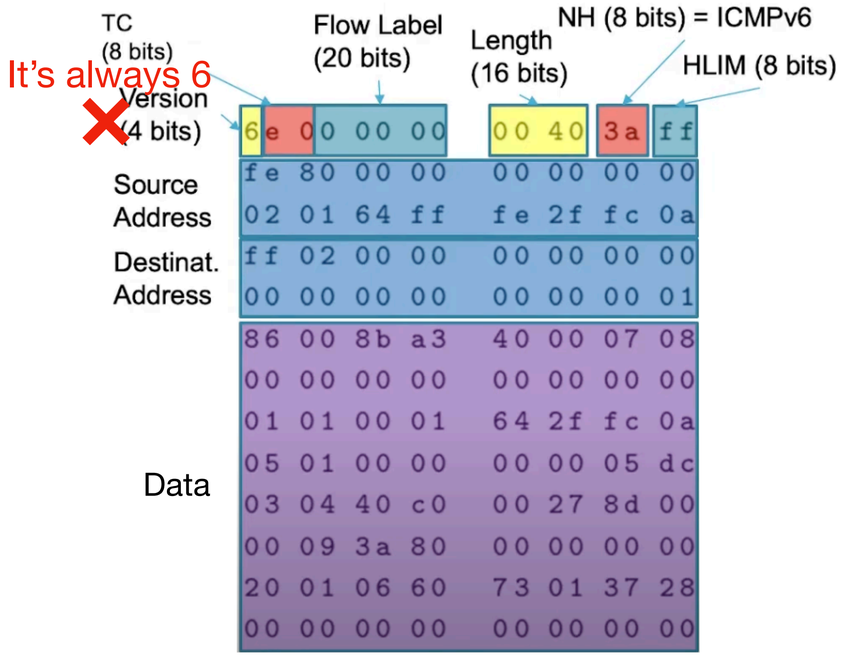


Header type, 011 indicates the use of IPHC

Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

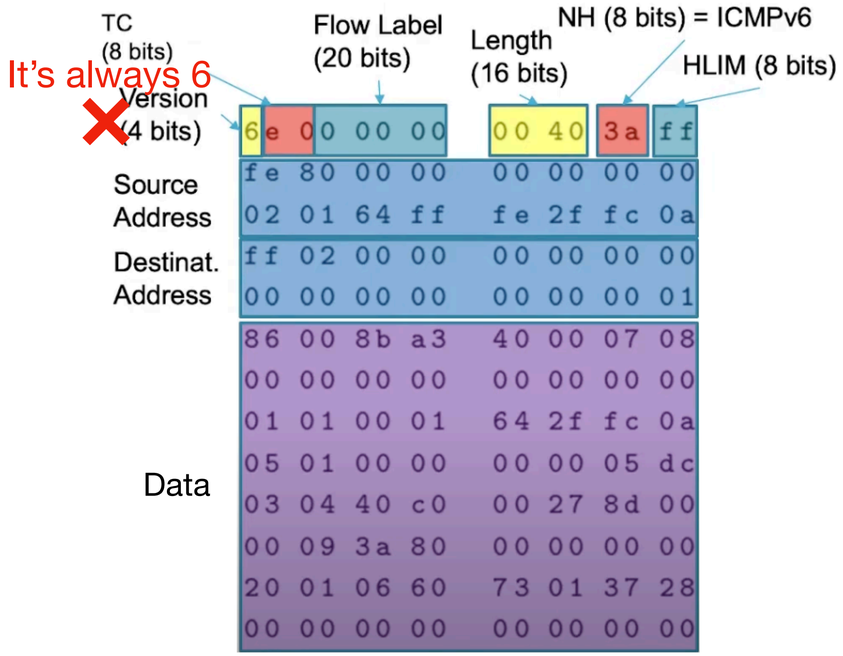
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Header Compression-scenario 1

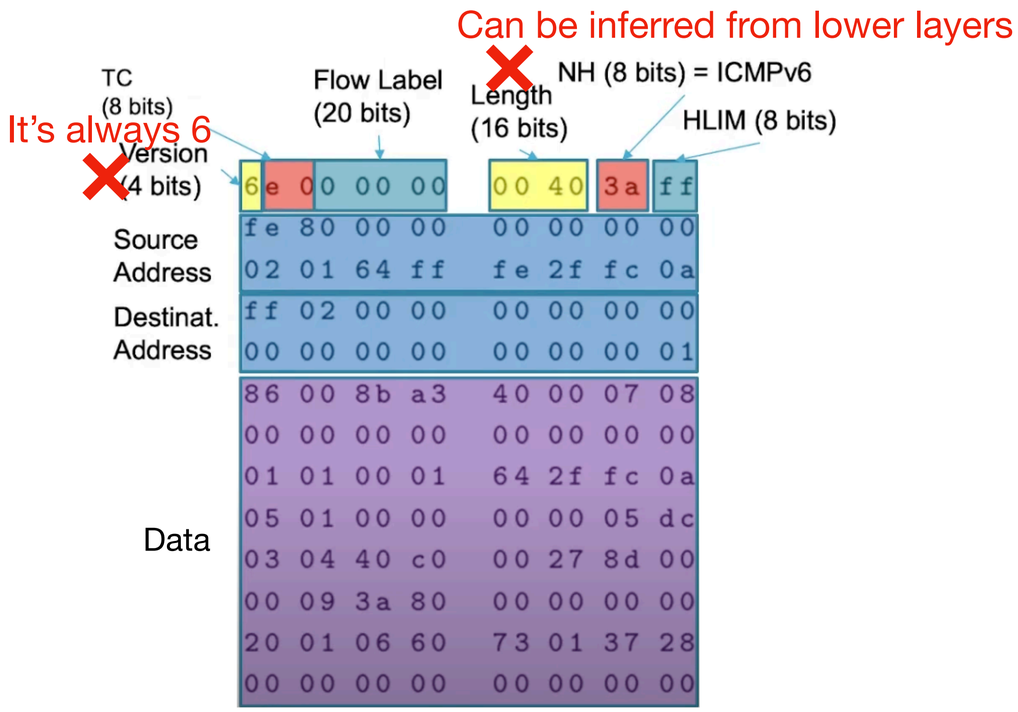


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Traffic/Flow bits, say whether the Traffic Class and/or Flow label were compressed

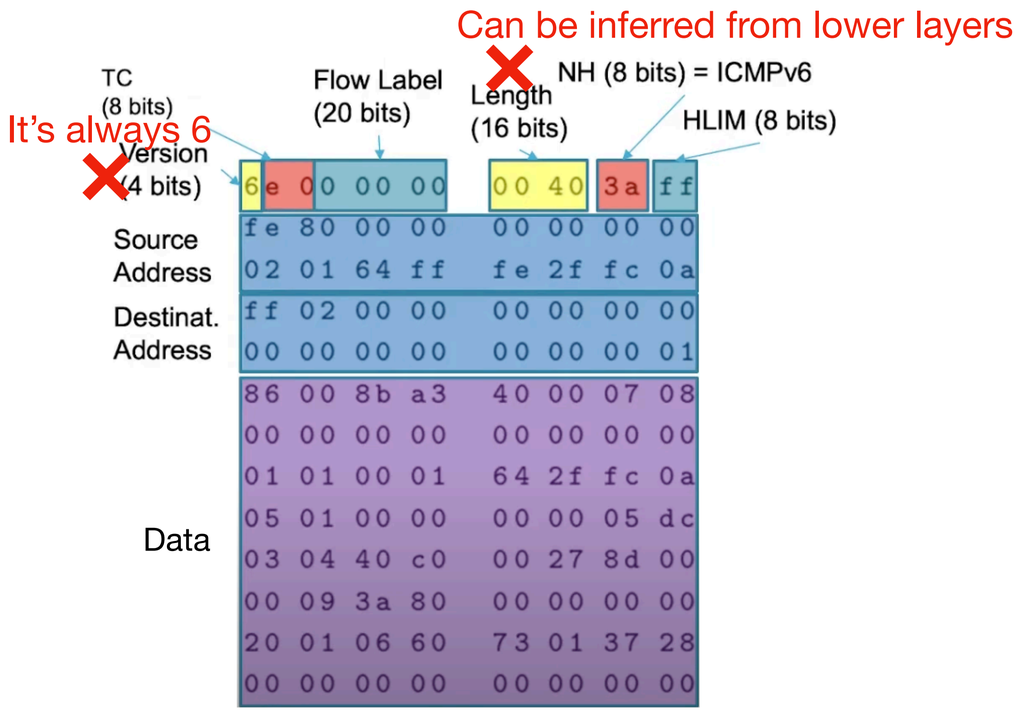
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Header Compression-scenario 1

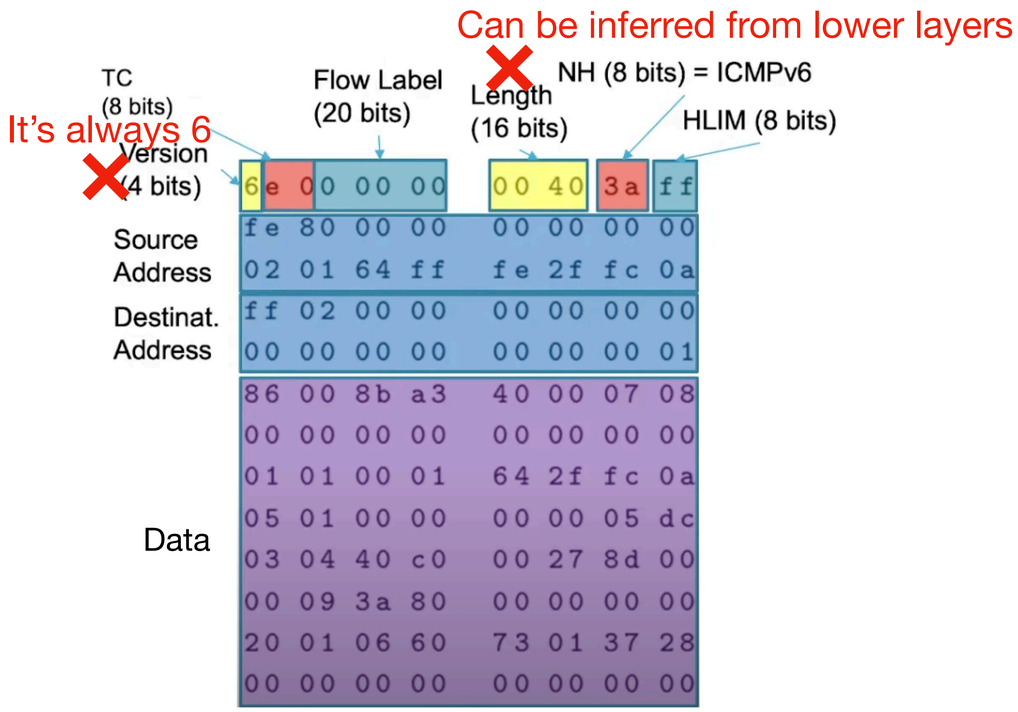


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Says whether the next header is compressed or not

Header Compression-scenario 1

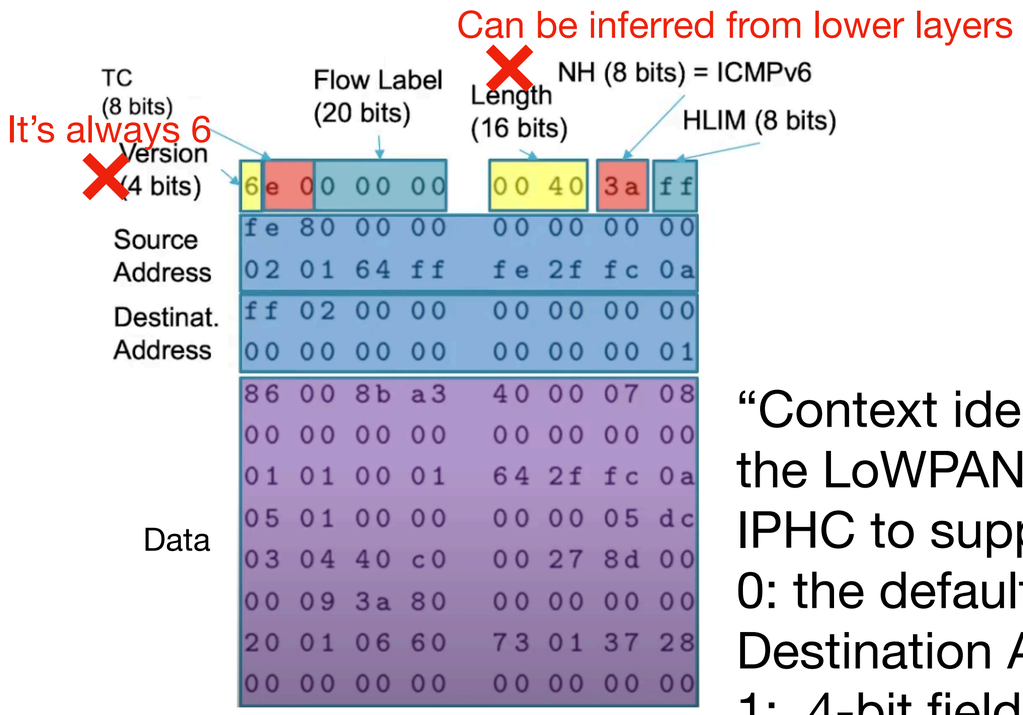


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

00: Hop Limit field carried in-line
01: Hop Limit field compressed and the hop limit is 1
10: Hop Limit field compressed and the hop limit is 64
11: Hop Limit field compressed and the hop limit is 255

Header Compression-scenario 1

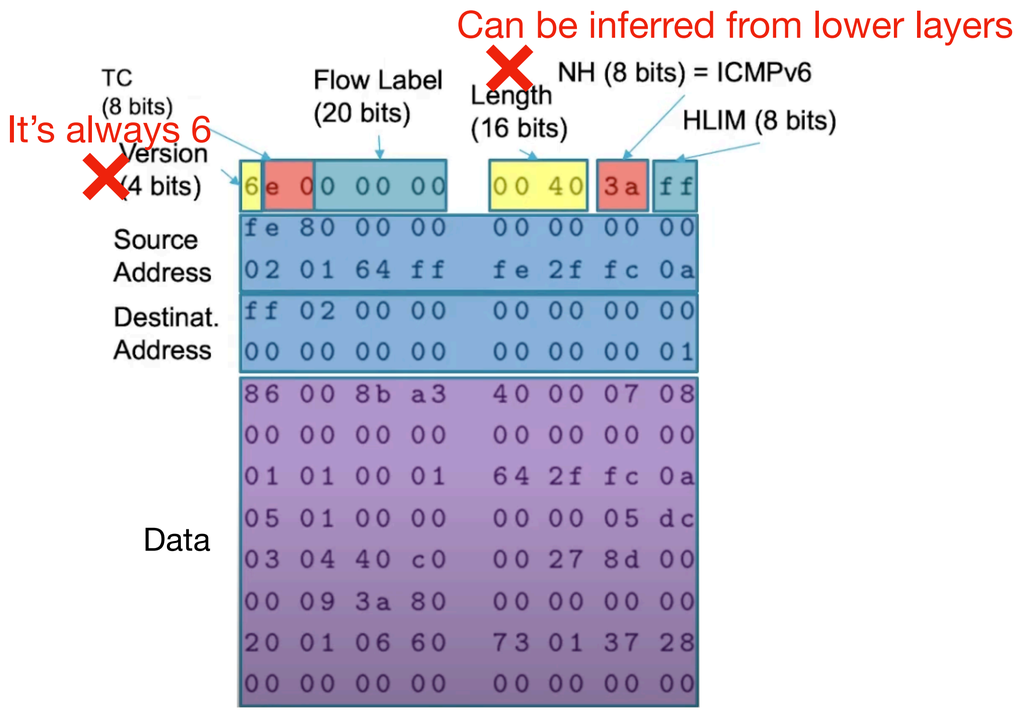


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

"Context identifier". Contexts act as shared states for all nodes within the LoWPAN. IPHC identifies the context using a 4-bit index, allowing IPHC to support up to 16 contexts simultaneously within the LoWPAN.
0: the default context may be used to compress Source and/or Destination Addresses
1: 4-bit fields follow the IPHC encoding to indicate one of the 16 contexts

Header Compression-scenario 1

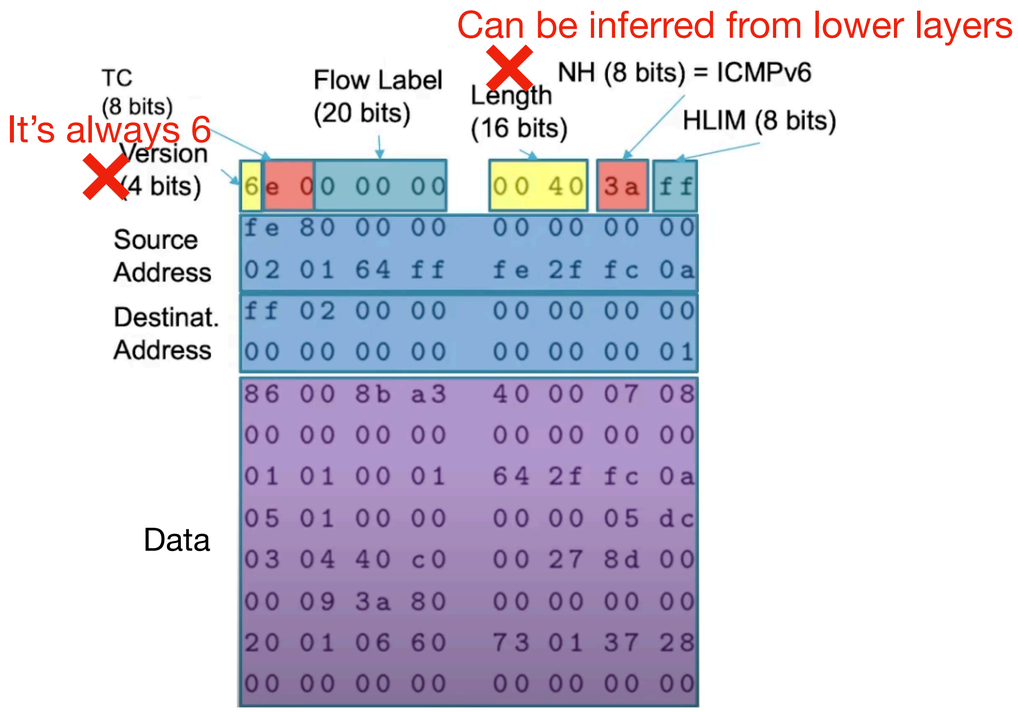


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Source Address compression:
0: stateless compression: no shared context
1: stateful compression: A shared context table is used to map context identifiers

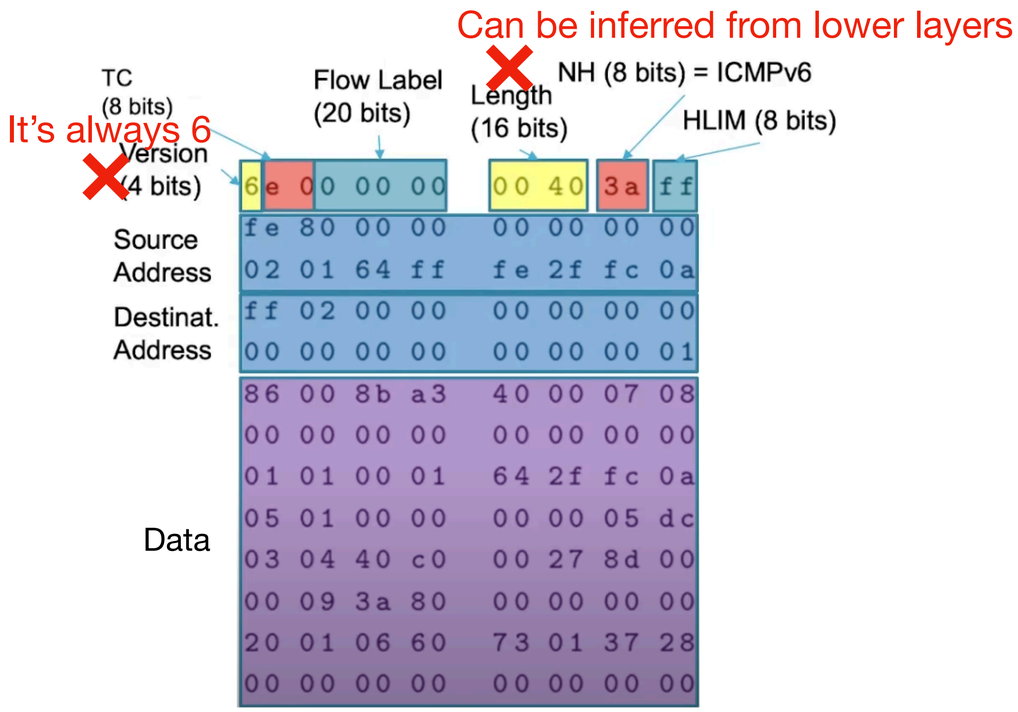
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Header Compression-scenario 1

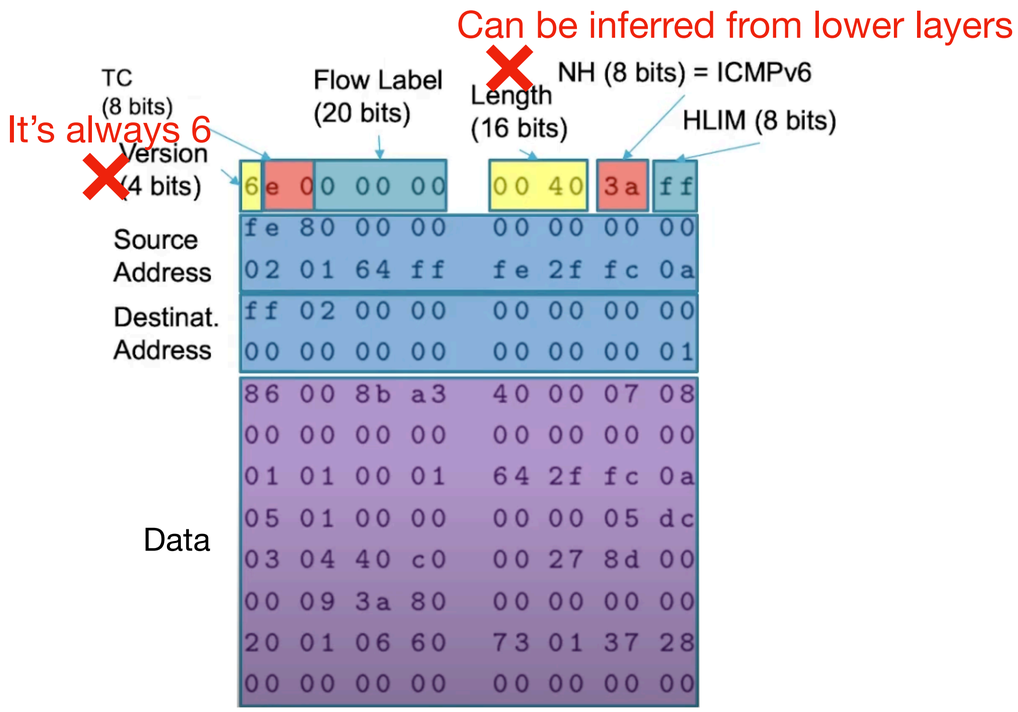


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

The Source Address Mode (SAM) indicates whether the full Source Address is carried inline, depending on the value of SAC

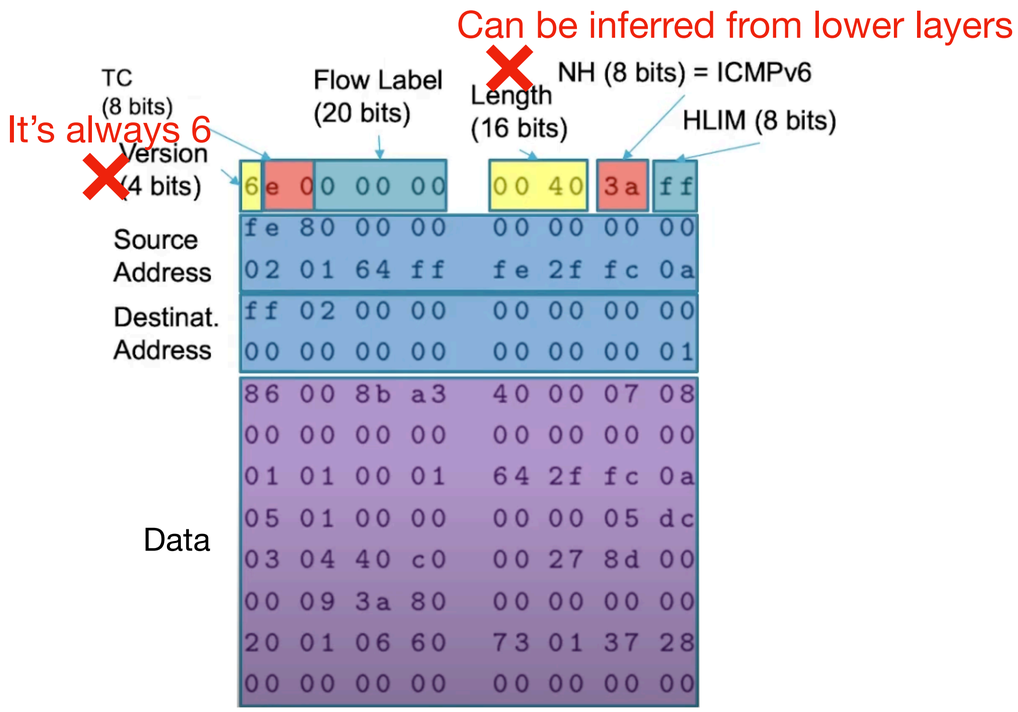
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

Header Compression-scenario 1

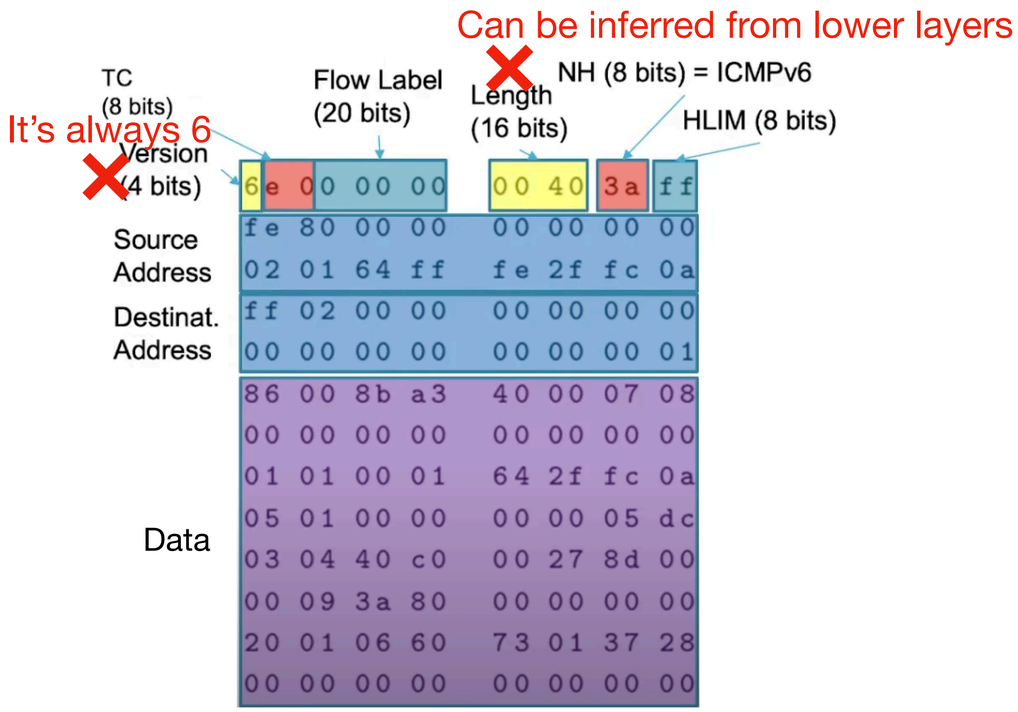


Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	

M: multicast mode
DAC (Destination access Compression) and DAM (Destination Access Mode) are similar to SAC and SAM, for the destination address.

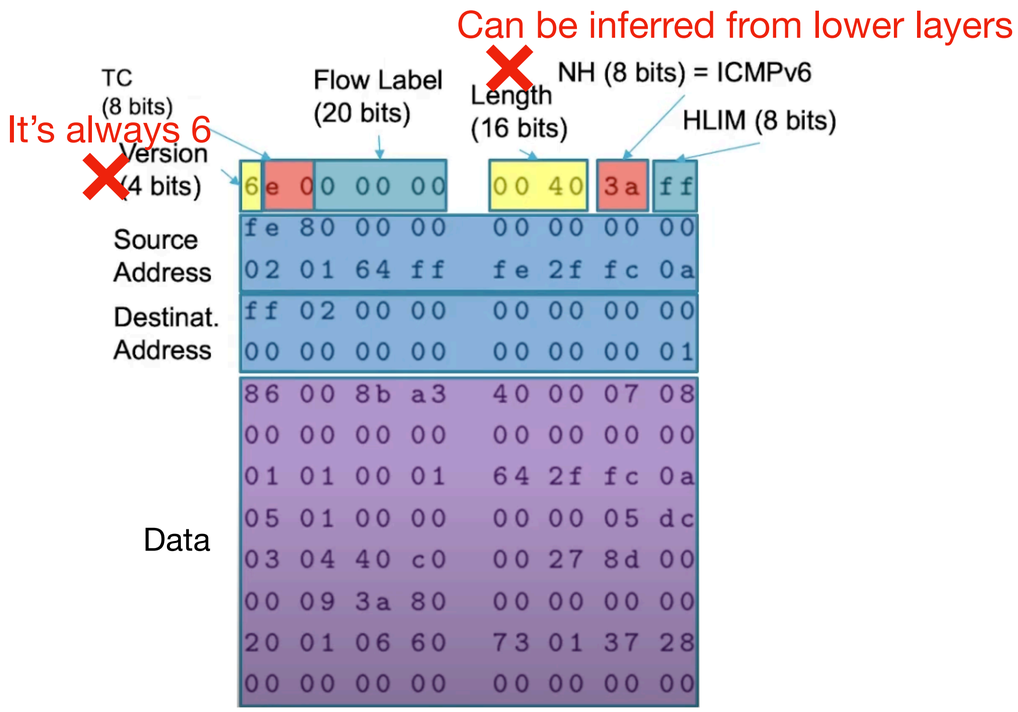
Header Compression-scenario 1



Compresses 6lowpan header IPHC

0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	
Uncompressed IPv6 fields							

Header Compression-scenario 1

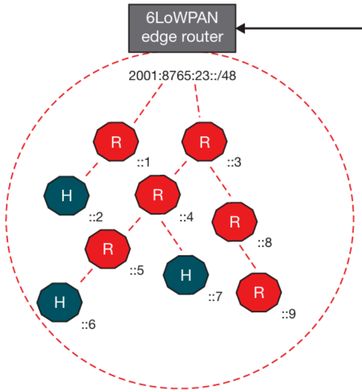


Compresses 6lowpan header IPHC

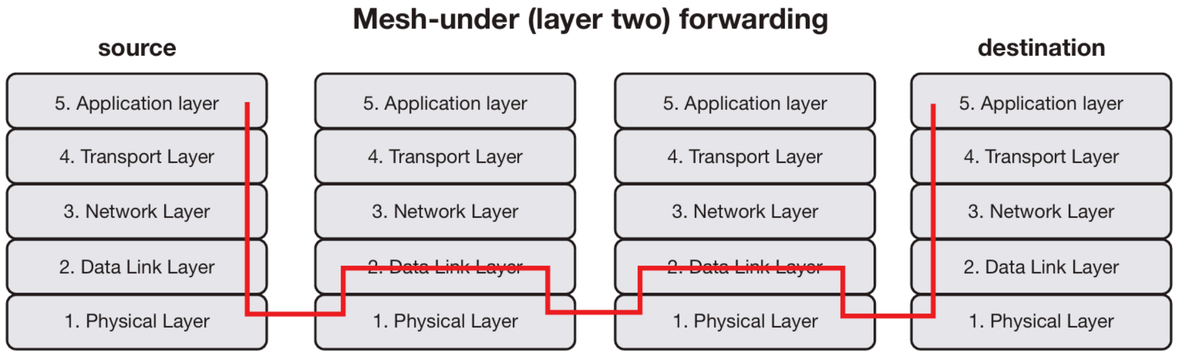
0	1	2	3	4	5	6	7
Dispatch field: 011			TF		NH	HLIM	
CID	SAC	SAM		M	DAC	DAM	
Uncompressed IPv6 fields							

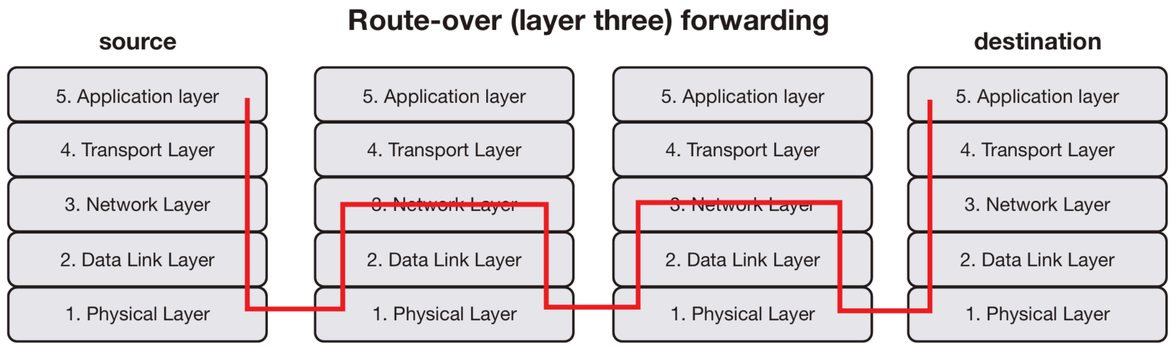
IPHC on a Link-Local Multicast Address with NH = ICMPv6: can compress 40 bytes IPv6 headers into 5 bytes, 2 bytes for the IPHC header, and 3 for the non-compressed IPv6 header fields

Routing



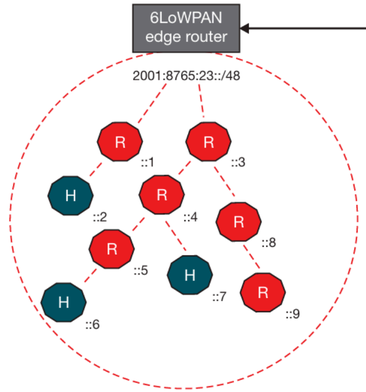
- 6LoWPAN has two routing modes, depending on what layer the routing mechanism is located:
  - Mesh-under.
    - Uses link layer addresses (IEEE 802.15.4 MAC) to forward data.
    - Mostly used in local networks where the only IP router is the edge router.
  - Route-over.
    - routing takes place at the IP level.
    - Used in larger and more powerful and scalable networks, where every router must implement all features supported by a normal IP router.



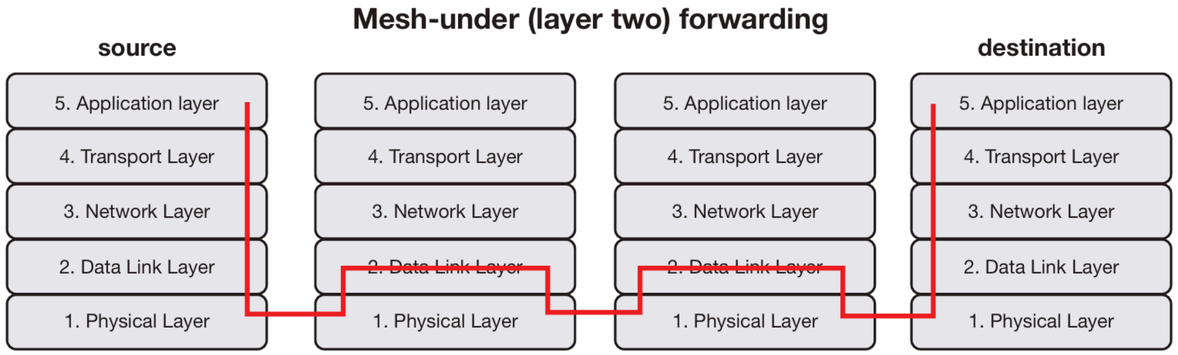


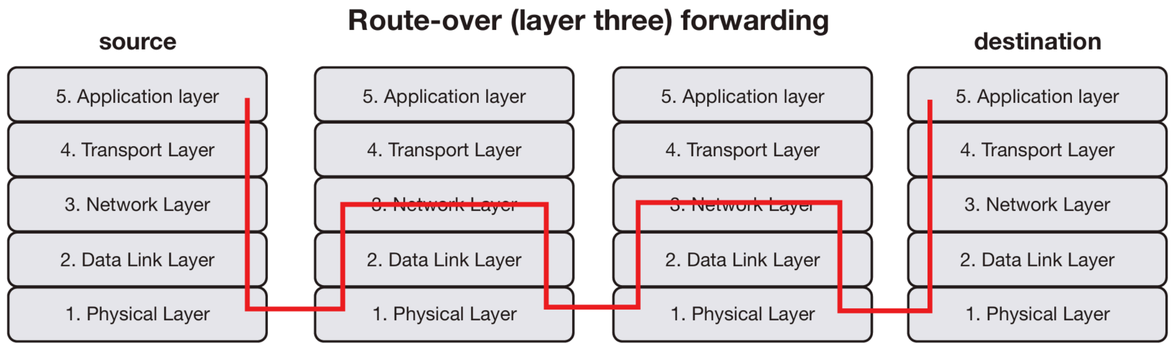
Routing

- 6LoWPAN has two routing modes, depending on what layer the routing mechanism is located:
  - Mesh-under.



If mesh-under routing is performed, the Mesh Addressing Header is attached.
Enables layer 2 multi-hop forwarding without requiring each node to reassemble and reprocess the IPv6 packet.





RLP (at a very high level)

- RLP (Routing Protocol for Low-power and Lossy Networks) is the most widely used routing protocol for route-over 6LoWPAN networks.

- It is a Distance-Vector Routing protocol that creates a routing tree called **Destination Oriented Directed Acyclic Graph (DODAG)**, typically rooted at the gateway or an edge router.

- Each node in the network is assigned a **rank** that represents its “distance” from the root.

- Ranks are assigned to nodes based on the **Objective Function** (OF) that the routing wants to optimise (e.g., number of hops [MIN], energy consumption [MIN], reliability - avoiding lossy links [MAX], etc).

Fragmentation

- In order to enable the transmission of IPv6 frames over IEEE 802.15.4 links, the IPv6 frames might need to be divided into several smaller segments (fragmentation).
  - In fact, although 6LoWPAN header compression can spare many bytes, the payload of the frames is still very small in IEEE 802.15.4.
- If fragmentation is performed, the fragmentation headers are generated to reassemble the packets in the correct order.
- Remember: IPv6 routers do not perform fragmentation, which is completely handled at the source side.

Fragmentation - routing modes

Mesh-under

- Intermediate nodes are not effected by IPv6 packets fragmentation. They perform link-layer packet forwarding.

Route-over

- Intermediate nodes process packets at layer 3 (IP).
- When the IPv6 packet's header is fragmented, it is split across multiple fragments.
- Fragments have to be **reassembled at each hop** before being processed.
  - Each intermediate node has to wait for all the fragments to reach, reassemble, inspect, and re-fragment them before routing them again (IPv6 is terminated at each node).
  - The fragmentation header includes the Datagram Size, Datagram Tag, and Datagram Offset fields to identify the fragment, and be able to re-assemble the entire IPv6 packet.

6.4 Application protocols

Application layer protocols in IoT

- Two most common application layer protocols in IoT:

- **MQTT** (**Message Queuing Telemetry Transport**) is a lightweight, publish-subscribe, machine-to-machine protocol for message queuing service.
  - designed for connecting servers with resource constrained, limited bandwidth devices (i.e., IoT devices)
  - TCP-based

- **CoAP** (**Constrained Application Protocol**) UDB-based Internet application protocol designed for constrained devices that need to communicate over a constrained network (low-power, lossy networks)

MQTT (1)

- MQTT uses a publish/subscribe pattern to decouple the message **sender** (**publisher**) from the message **receiver** (**subscriber**).
- A third component (the **broker**) handles the communication between publishers and subscribers.
- The broker filters all incoming messages from publishers and distributes them correctly to subscribers.
- The broker decouples the publishers and subscribers as follows:
  - **Space decoupling**: the published and subscribers are not aware of each other's network location and do not exchange information such as IP addresses or port numbers
  - **Time decoupling**: the publisher and subscriber do not run or have network connectivity at the same time
  - **Synchronisation decoupling**: publisher and subscribers can send or receive data without interrupting each other.

MQTT components

- An **MQTT client** is any device (from a server to a microcontroller) that runs a MQTT library. If a client is sending a message, it acts as a publisher, and if it receives a message, it acts as a subscriber (any device communicating with MQTT over a network is a MQTT client)

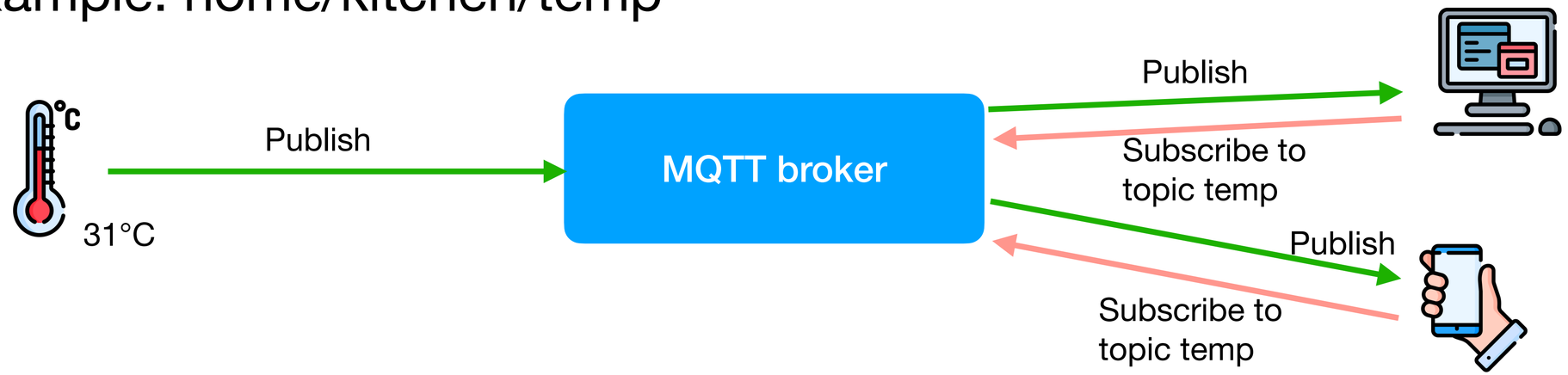
- An **MQTT broker** is the backend system which coordinates messages between clients.
  Its roles include:
  - receiving and filtering messages
  - identifying clients subscribed to each message and sending them the messages
  - authorising and authenticating MQTT clients
  - Passing messages to other systems for further analysis.

MQTT components

- MQTT clients and brokers begin communicating using an **MQTT connection.**
  - Clients initiate the connection by sending a CONNECT message to the MQTT broker.
  - The broker confirms that a connection has been established by responding with a CONNACK message.
  - Clients never connect with each other, only with the broker.
- Brokers can be:
  - managed brokers, i.e., services that let the user use their hosted brokers in their servers for the user's system.
  - Self-Hosted brokers require the users to install the broker on their server with a static IP. Open source implementation of MQTT brokers include mosquito e hivemq

MQTT (2)

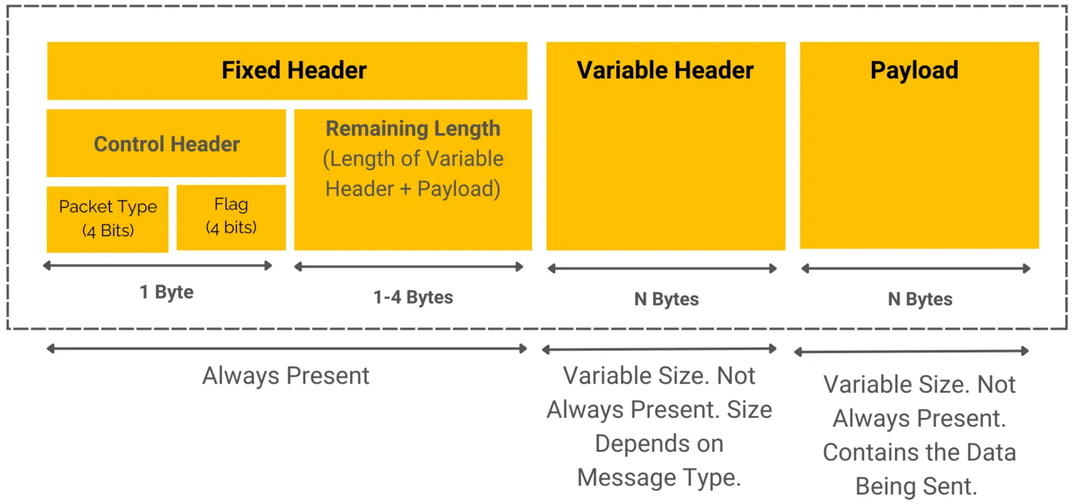
- Key feature of MQTT is **MQTT topics.** MQTT topics are keywords the MQTT broker uses to filter messages.
- Topic are organised hierarchically, similarly to a folder directory.
  - example: home/kitchen/temp



- It supports a **stateful** architectural approach, maintaining persistent **sessions.**
  - In IoT scenarios with unstable connectivity, MQTT allows clients to reconnect and resume communications without losing context.

MQTT control packets

- **CONNECT packet**:
  - Fixed header: identifies the packet as a CONNECT packet.
  - Variable header: contains protocol names, version and connection flags
  - Payload: includes client ID, username, password, etc.



- **PUBLISH packet**:
  - Fixed header: identifies as a PUBLISH packet
  - Variable header: contains **topic name** and packet identifier
  - Payload: actual message being published (e.g., 31°C)
- **SUBSCRIBE packet**:
  - Fixed header: identifies as a SUBSCRIBE packet
  - Variable header: contains packet identifier
  - Payload: list of topics to subscribe to and their QoS

CoAP (1)

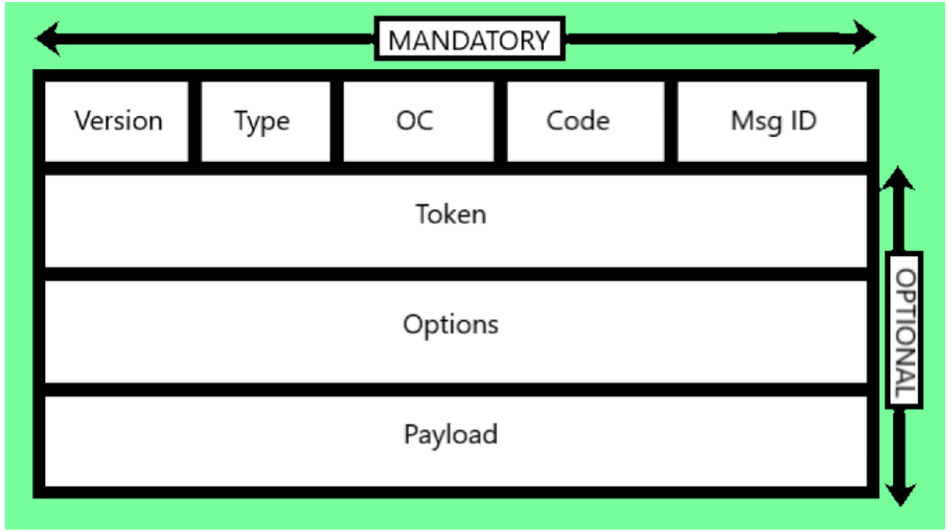
- CoAP follows a client/server model, enabling the client to request for services from server as needed. Servers respond to clients' requests.
- CoAP is **resource-oriented**: it treats various contents as resources, each uniquely identified by a **URI** (Uniform Resource identifier). Clients can request information about these resources to servers.
  - Resources are the information that different applications provide to their clients.
    - Can be images, videos, text, numbers, or any type of data.
  - Based on **REST**-style architecture, **stateless** (the necessary state to handle the request is contained within the request itself and server would not store anything related to the session. The client must include all information for the server to fulfil the request).

CoAP (2)

- Supports **methods** as in HTTP:
  - GET: used to retrieve resources information identified by the URI.
  - POST: creates a new resource.
  - DELETE: deletes the resources identified by the URI
  - PUT: updates and creates resources identified by the URI
- Has an extra option, used with GET:
  - OBSERVE: alerts the server to send every update about the resources to the client
- Uses UDP, allowing for asynchronous messaging.
  - No ack, helps preserve energy in resource-constrained devices.

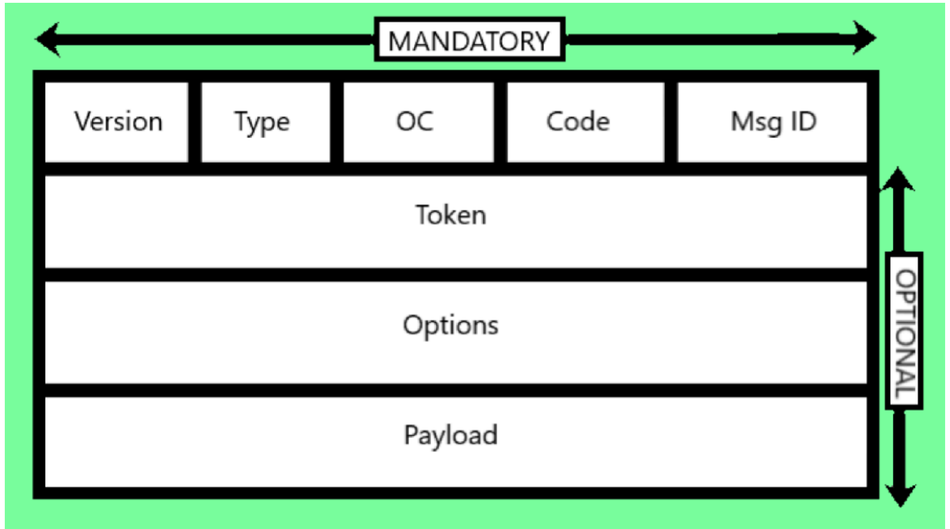
CoAP message format (1)

- **Version**: 2 bits. Represents version of CoAP protocol
- **Type**: 2 bits for 4 types of messages
  - request: 0: confirmable (expects ACK)
    1: non-confirmable
  - response: 2: ACK
    3: RESET - could receive msg but could not process it (e.g., context is missing)



- **Option Count**: 4 bits
- **Code**: 8 bits. Indicates the method and whether the message is a request or a response.
- **Message ID**: 16 bits

CoAP message format (2)



- **Token**: variable size, ranges between 0 to 8 bytes. Each request carries a token. It is a client-local identifier to match requests and responses
- **Options**: variable size, provide a flexible and compact way to include additional metadata or parameters in a message (as media type, lifetime of resources, etc)
- **Payload**: variable size. It is typically a representation of the requested resource or the result of the requested action.

MQTT simulation with paho clients

- https://mqtt.eclipseprojects.io/ This is an MQTT broker you can connect to.
- https://pypi.org/project/paho-mqtt/ install paho for an MQTT client.

Bibliography

- Hanes et al. ”IoT Fundamentals: Networking Technologies, Protocols, and Use Casesfor the Internet of Things”. Cisco Press
- Sthapit, Pranesh, and Jae-Young Pyun. "Station grouping strategy for minimizing association delay in IEEE 802.11 ah." IEICE Transactions on Communications 100.8 (2017): 1419-1427.
- Tian, Le, et al. "Wi-Fi HaLow for the Internet of Things: An up-to-date survey on IEEE 802.11 ah research." Journal of Network and Computer Applications 182 (2021): 103036.
- Khorov, Evgeny, et al. "A survey on IEEE 802.11 ah: An enabling networking technology for smart cities." Computer communications 58 (2015): 53-69.
- Culler, David, and Samita Chakrabarti. "6LoWPAN: Incorporating IEEE 802.15. 4 into the IP architecture." White paper (2009).
- Olsson, Jonas. "6LoWPAN demystified." Texas Instruments 13 (2014): 1-13. APA
- https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/Core-54/out/en/host.html
- https://lpccs-docs.renesas.com/tutorial-custom-profile-DA145xx/introduction.html
- https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/
- https://www.hivemq.com/blog/mqtt-packets-comprehensive-guide/