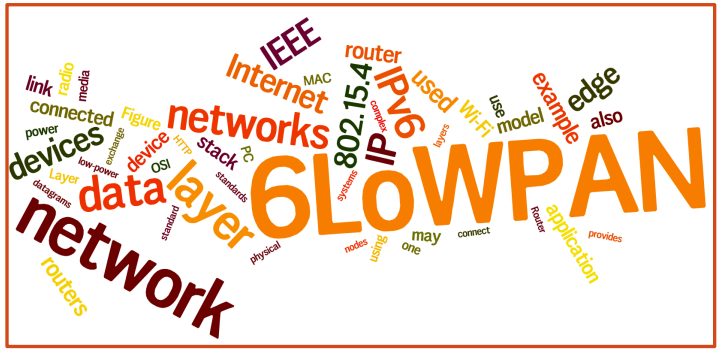


The Internet of Things:
IP-based Network Layer Solutions

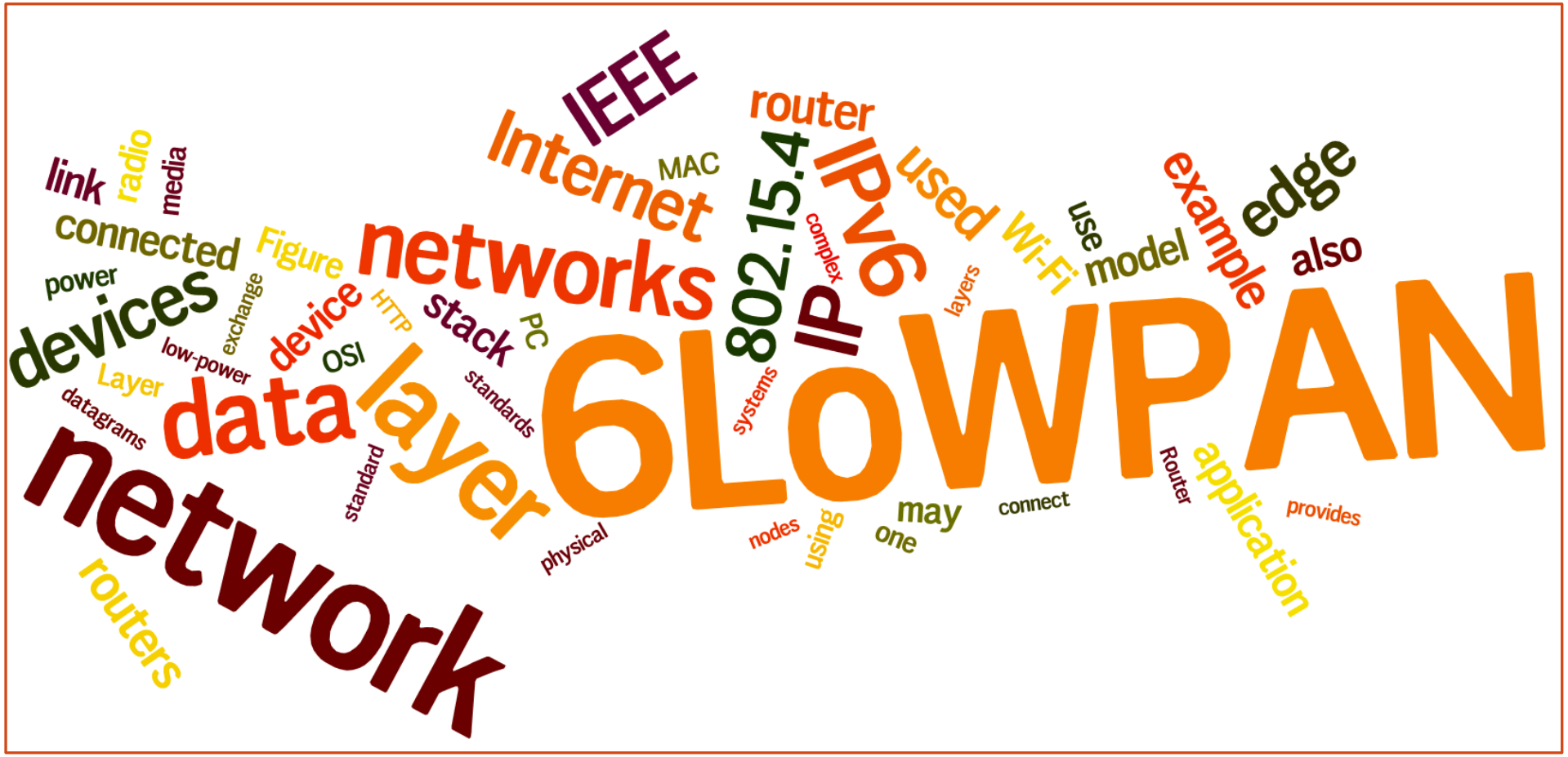
Course website: http://site.unibo.it/iot

Prof. Luciano Bononi
luciano.bononi@unibo.it

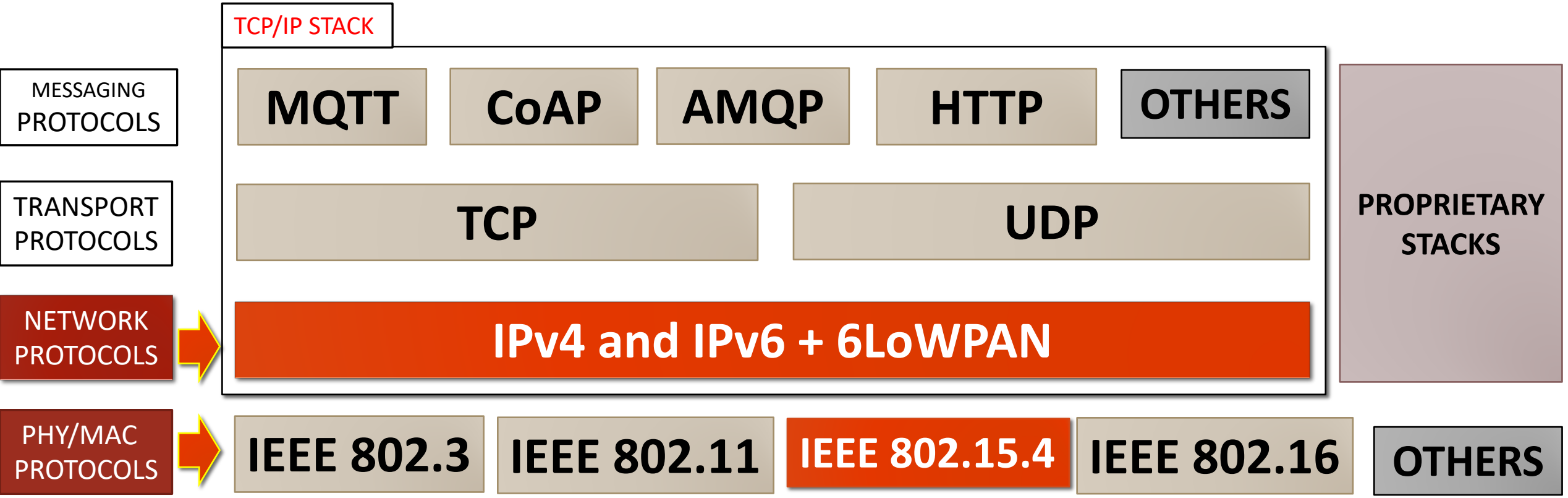
Prof. Marco Di Felice
marco.difelice3@unibo.it

MASTER DEGREE IN COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, UNIVERSITY OF BOLOGNA, ITALY

IoT Protocol Stack

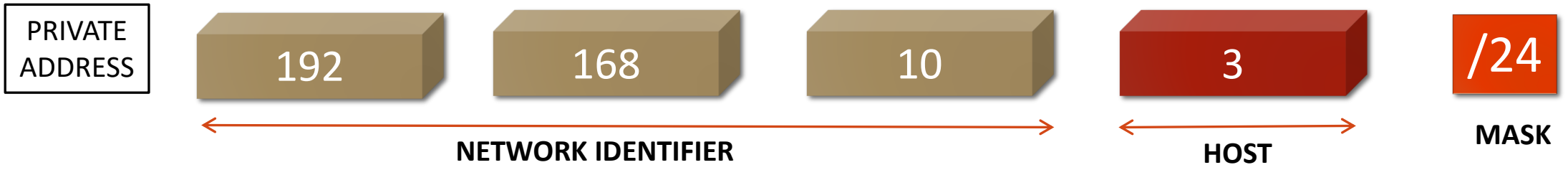


IoT Protocol Stack



IPv4 Protocol

- **IP version 4** (IPv4)
  - First version deployed by the ARPANET project in 1983
  - Uses **32-bit network addresses** (address space → 4294967296 values).
  - IPv4 can be **public** (i.e. routable over the Internet) or **private**
  - Each IPv4 address contains **two parts**: the (i) network identifier and the (ii) host identifier. The **network mask** indicates the number of bits (over the 32) used to represent the network identifier.

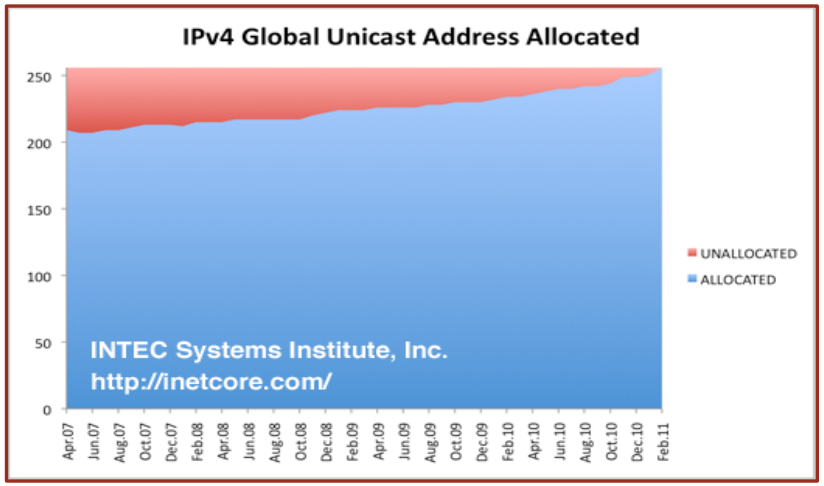


IPv6 Protocol

❑ IP version 6 (IPv6)

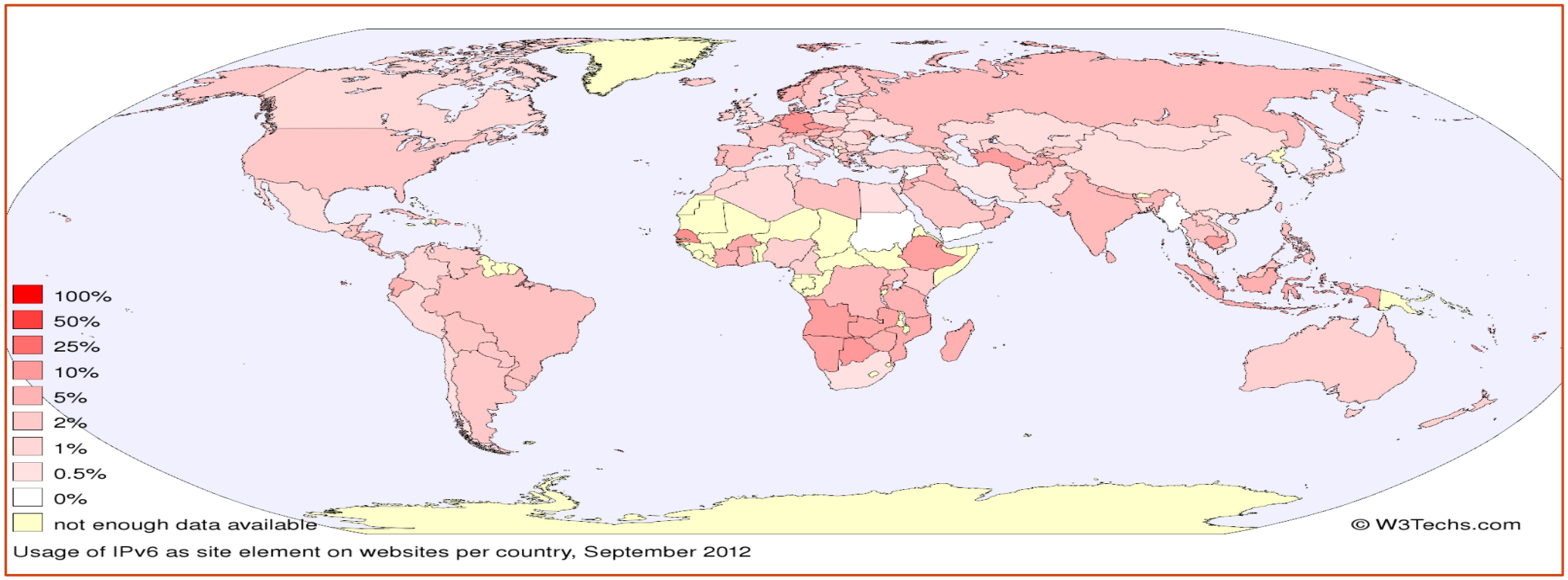
- ✧ Developed by the Internet Engineering Task Force (1998).
- ✧ Replace IPv4 and address the IPv4 address exhaustion problem.
- ✧ Additional routing functionalities (not included in IPv4).
- ✧ Not compatible with the IPv4 protocol.

- ❑ **The migration process** to IPv6 involves: network infrastructures, routers, applications
- ❑ Complete migration expected by 2025



IPv6 Protocol

❑ IP version 6 (IPv6) adoption worldwide



Usage of IPv6 as site element on websites per country, September 2012

IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)

1. Extended **addressing capabilities**

IPv4 address: 32 bit, IPv6 address: 128 bit → **2^{128} combinations available**!

3FFE:085B:1F1F:0000:0000:0000:00A9:1234

8 groups of 16-bit hexadecimal numbers separated by ":"

Leading zeros can be removed → 3FFE:85B:1F1F::A9:1234

IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)
  1. Extended **addressing capabilities**

Three types of IPv6 addresses:

- **Unicast**: one-to-one communication
- **Multicast**: one-to-many communication
- **Anycast**: one-to-a-group, and a single destination is chosen
- **Broadcast**: not supported

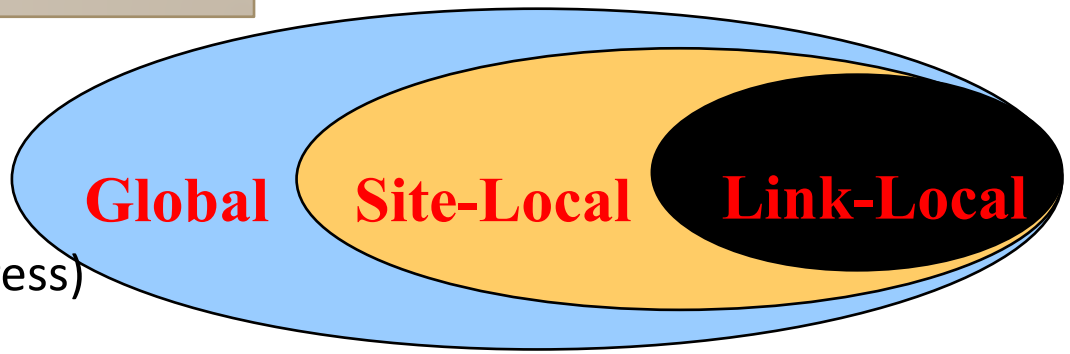
IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)

1. Extended **addressing capabilities**

A network interface can have multiple **addresses**

LINK-LOCAL ADDRESSES



- Start using a link-local prefix **FE80::/10**
- Contain the interface identifier (e.g. MAC address) in the modified EUI-64 format.
- Can be used to reach the neighboring nodes attached to the same link
- IPv6 routers must not forward packets having link-local source/destination
- All IPv6 enabled interfaces have a link-local unicast address.

IPv6 Protocol

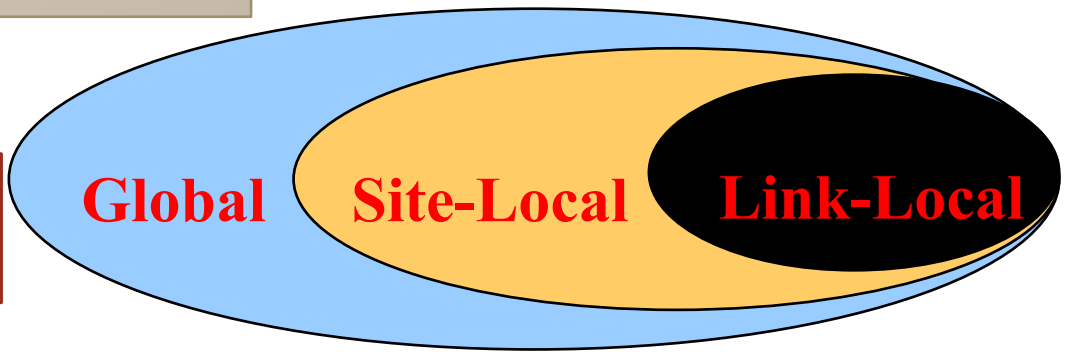
- Novel **features** of the IPv6 protocol (compared to IPv4)

1. Extended **addressing capabilities**

A network interface can have multiple **addresses**

SITE-LOCAL ADDRESS

- ✧ Start using a link-local prefix **FC00::/7**
- ✧ Similar properties as IPV4 **private** addresses



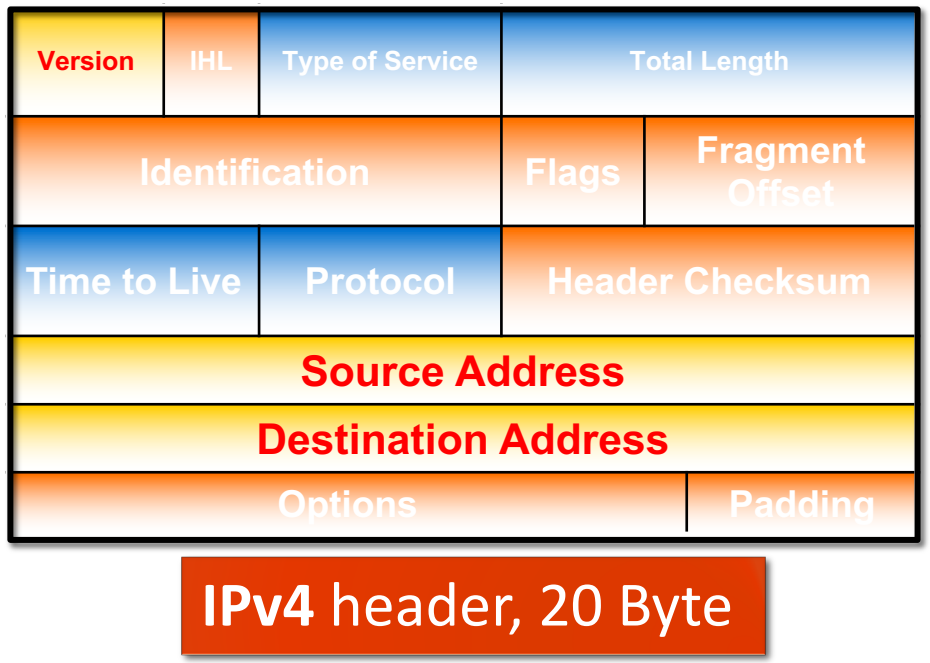
GLOBAL ADDRESS

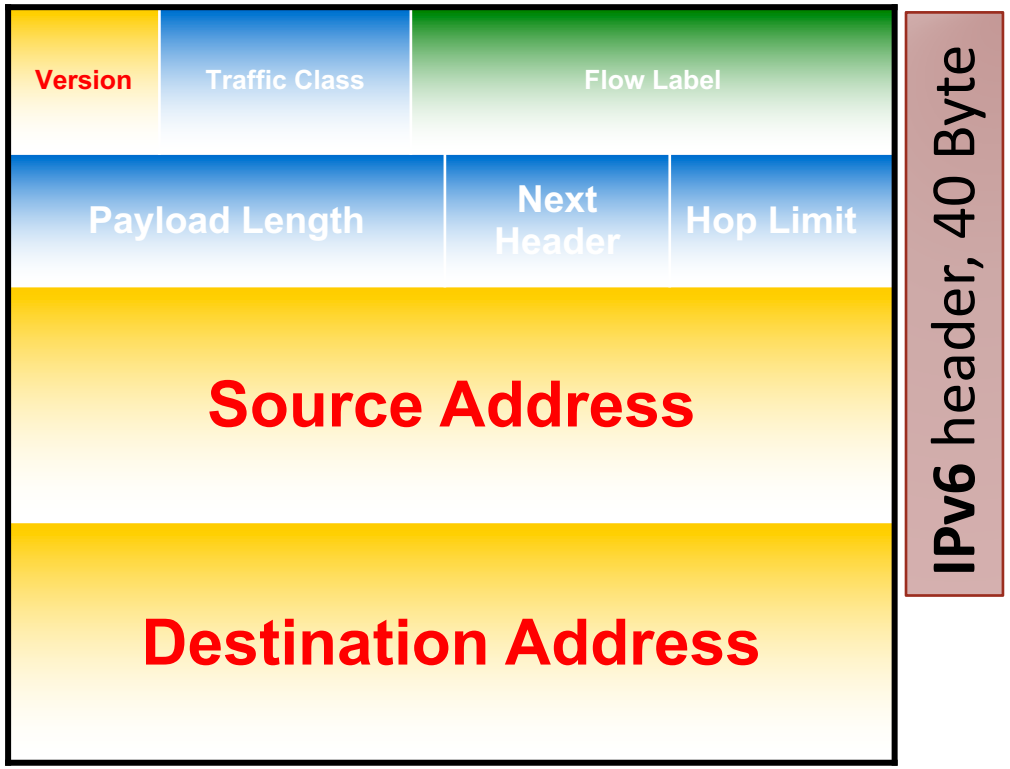
- ✧ Can be used to route IP datagrams over the Internet
- ✧ Variable prefix, defined from router advertisements. Some IP addresses can be reserved.

IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)

2. IP Header **re-newed**

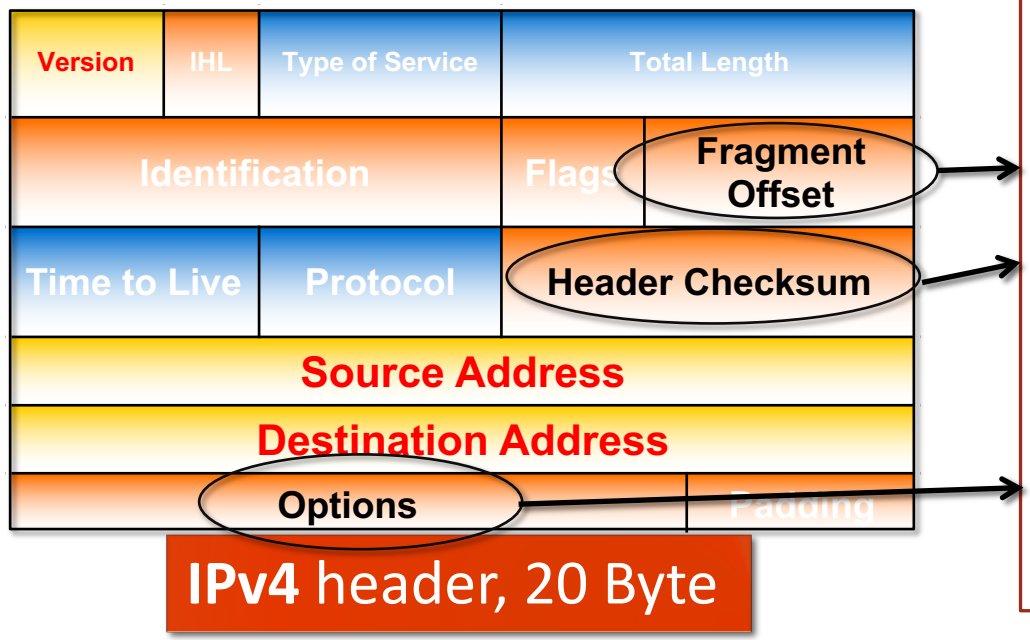




IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)

  2. IP Header **re-newed**



Fields removed in the IPv4 header:

- **Checksum** → replicated in MAC and TSP header, not needed at the IP layer.
- **Fragmentation** → fragmentation is performed by end-points, while might not be supported by routers.
- **Options** → replaced by pointer to next header extension (next header).

IPv6 Protocol

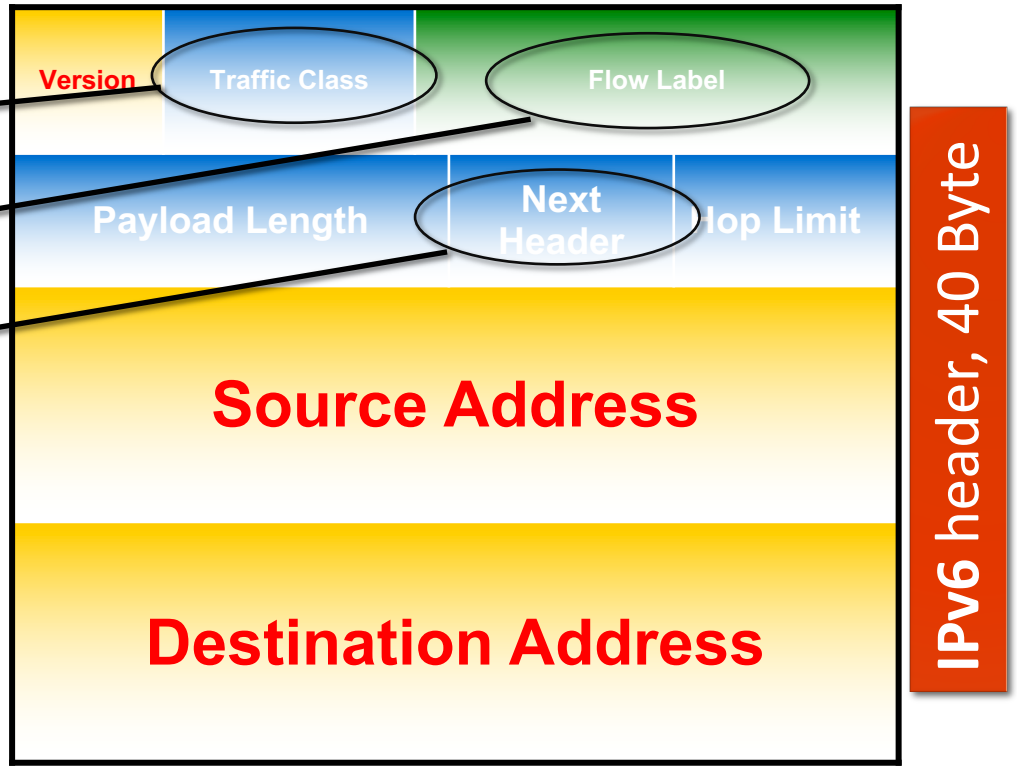
- Novel **features** of the IPv6 protocol (compared to IPv4)

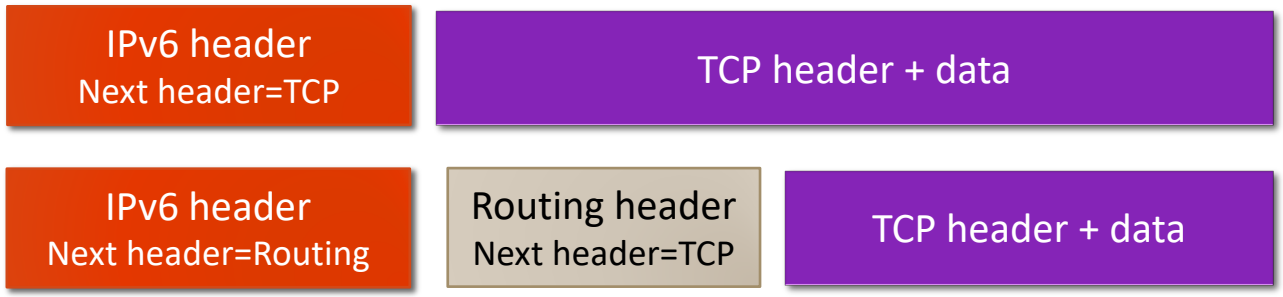
2. IP Header **re-newed**

Identify possible **QoS** requirements

Identify a **source-destination traffic flow**

Pointer to next **header** extension (optional)





IPv6 Protocol

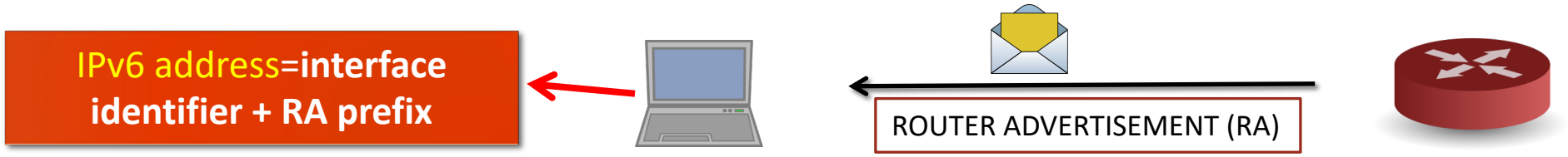
- Novel **features** of the IPv6 protocol (compared to IPv4)

  3. IP Address **assignment** process, three ways

✧ **Manual** configuration → like using the "**ifconfig**" utility

✧ **Stateful** configuration → using **DHCPv6** protocol

✧ **Stateless autoconfiguration → no DHCP,** IPv6 nodes can connect to a network and **automatically generate global IPv6 addresses** without the need for manual configuration or help of a server.



IPv6 Protocol

- Novel **features** of the IPv6 protocol (compared to IPv4)

  3. IP Address **assignment** process, three ways

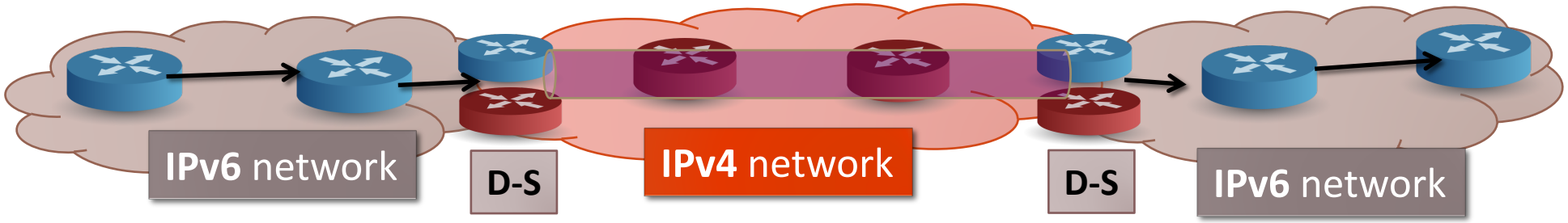
✧ **Manual** configuration → like using "**ifconfig**" utility

✧ **Stateful** configuration → using **DHCPv6** protocol

✧ **Stateless autoconfiguration → no DHCP,** IPv6 nodes can connect to a network and **automatically generate global IPv6 addresses** without the need for manual configuration or help of a server.

IPv6 Protocol

- Managing transition from IPv4 to IPv6
  - **Dual-stack** approach
    Some routers will support both IPv4 and IPv6 protocols
  - **GRE Tunnelling** approach
    Communication tunnels enable communication between IPv6 subnetworks over IPv4 links



IPv6 Protocol and the IoT

- Benefits of using IPv6 protocols on **IoT scenarios**:
  - Address/manage/access **any IoT device from the Internet.**
  - Easily connect to other IP networks without the need of **translation gateways** or **proxies**.
  - Use **well-known socket APIs** for the deployment of network application.
  - Easily **re-use tools** for managing, commissioning and diagnosing IP-based networks.
  - **Leverage on the addressing capability of the IPv6 protocol.**

IPv6 Protocol and the IoT

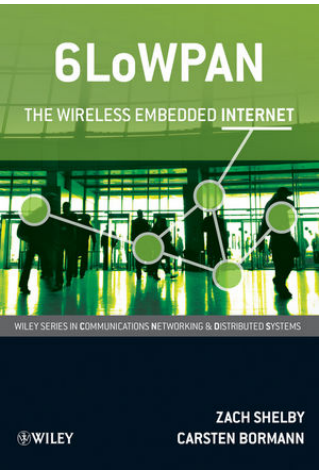
- At the same time, supporting IPv6 over IoT scenarios present several challenges:
  - **IPv6 datagrams are not a natural fit** for IEEE 802.15.4 networks
    - MTU size of an IEEE 802.15.4 frame is 127 bytes, while the minimum IPv6 frame size is 1280 bytes;
    - The IPv6 header size (40 bytes) can occupy 1/3 of the MTU
  - IPv6 assumes that a **link is a single broadcast domain**, while the assumption does not hold in multi-hop wireless sensor networks.
  - IPv6 includes **optional support for IP security** (IPsec), authentication and encryption but these techniques might be too complex for IoT-devices.

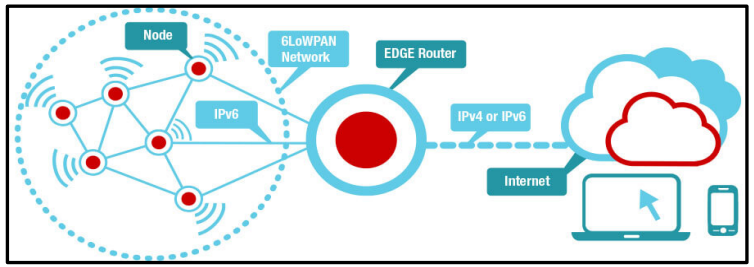
IPv6 Protocol and the IoT

- Worst case scenario calculations.
  - Maximum **frase size** in IEEE 802.15.4 → 127 bytes
  - Reduced by the **max frame header** (25 bytes) → 102 bytes
  - Reduced by the **highest link layer security** (21 bytes) → 81 bytes
  - Reduced by **standard IPv6 header** (40 bytes) → 41 bytes
  - Reduced by **standard UDP header** (8 bytes) → 33 bytes
  - **Only 33 bytes left for data payload!**

6LoWPAN

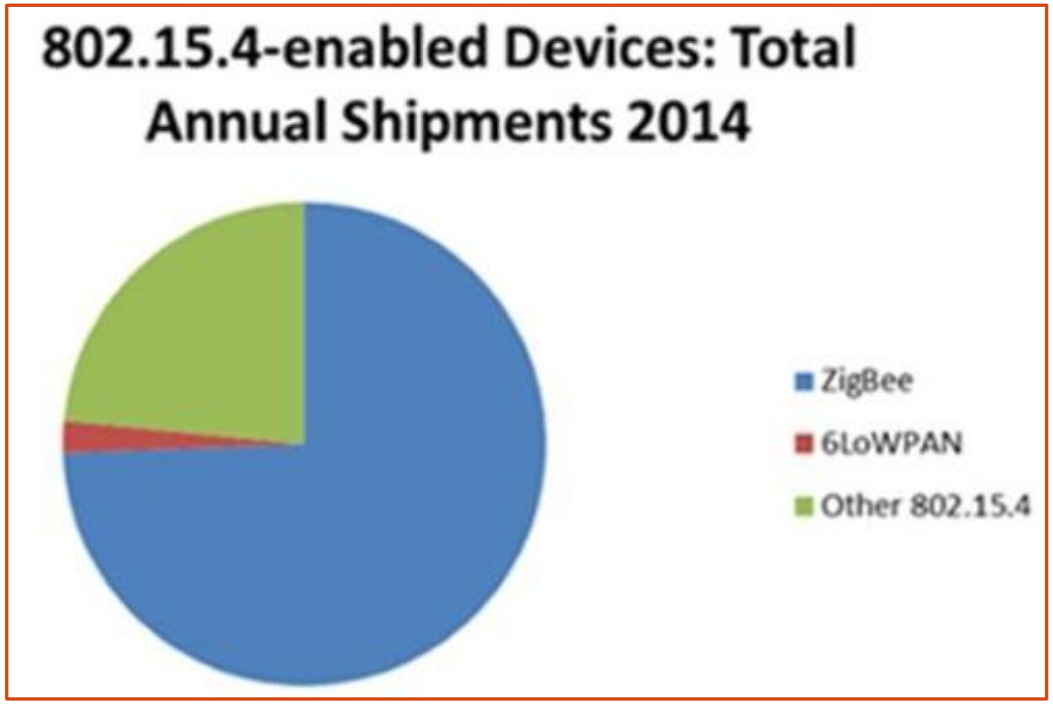
- Set of standards defined by the **Internet Engineering Task Force** (IETF) enabling the efficient use of IPv6 over low-power, low-rate wireless networks on simple embedded IoT devices. It provides:
  - ✧ A novel **Adaptation Layer;**
  - ✧ Several **optimization** of IPv6 functionalities.

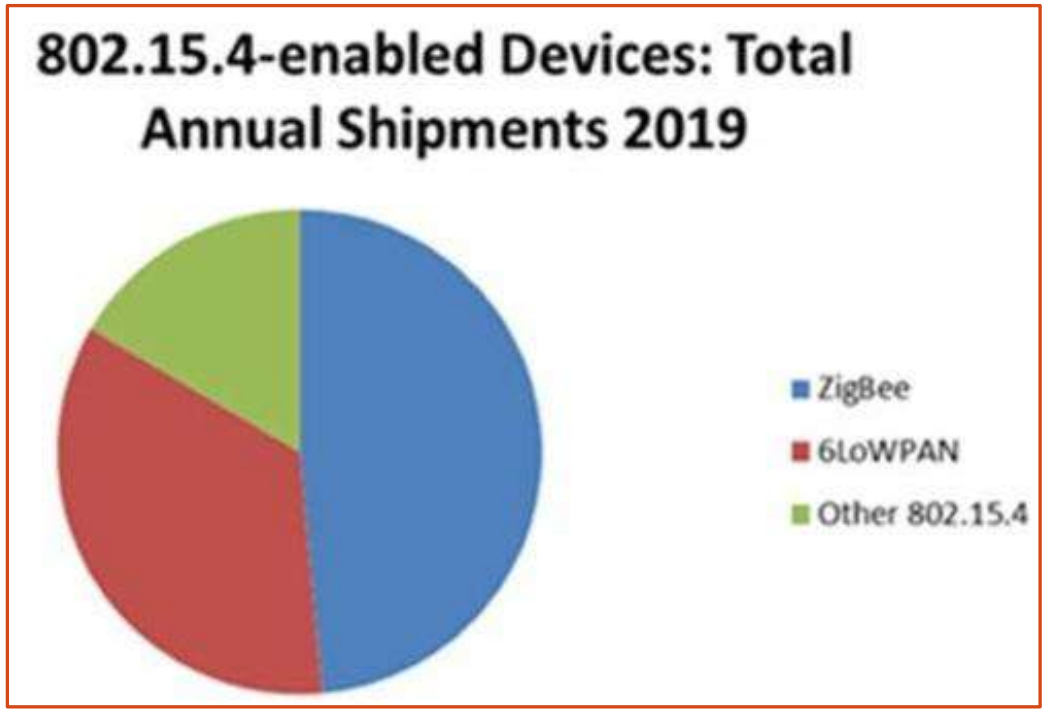




- ✧ RFC 4919 (first specification, 2007)
- ✧ RFC 4944 (auto-configuration)
- ✧ RFC 6282 (header compression)
- ✧ RFC 7400 (header compression)
- ✧ …

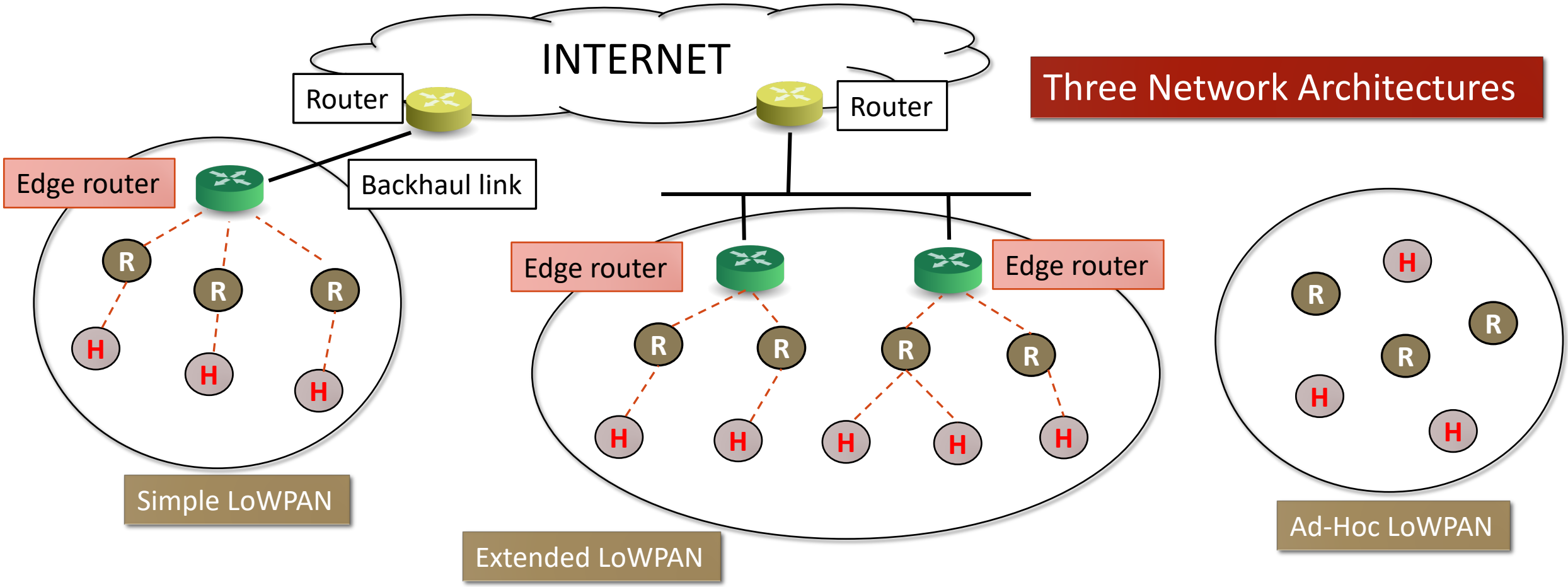
6LoWPAN MarketShare



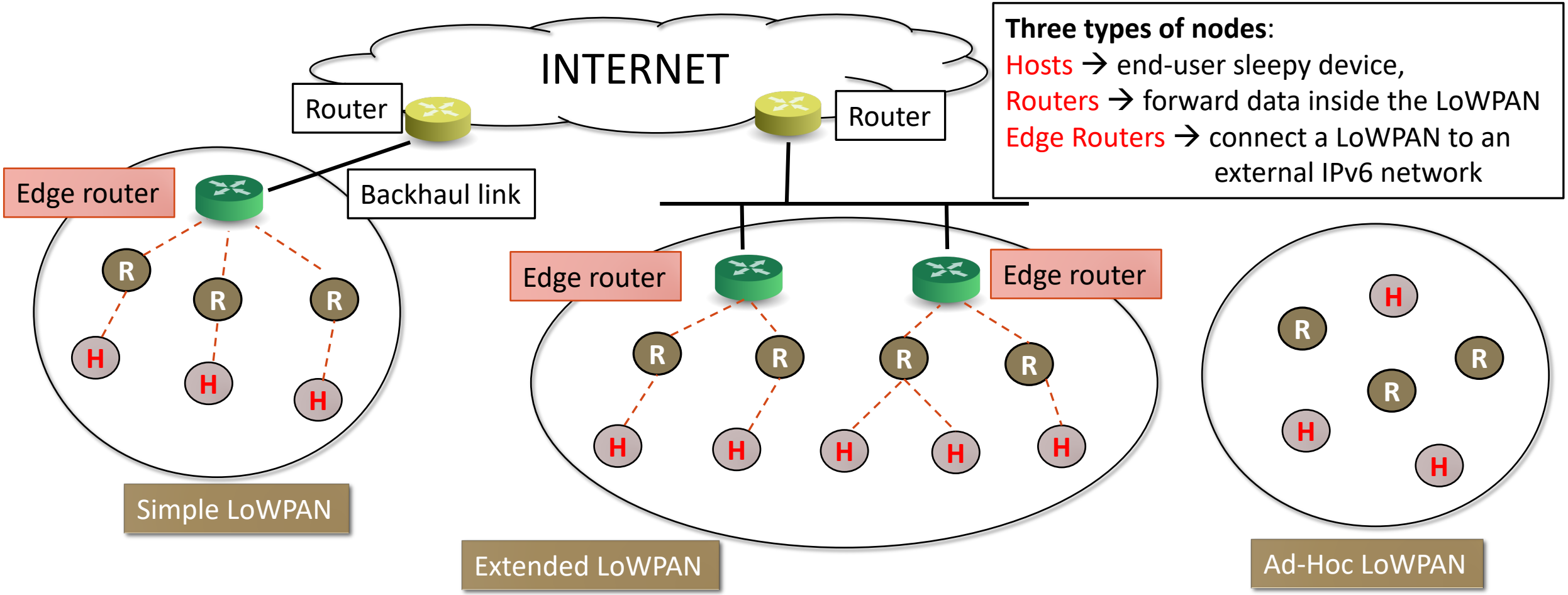


Source: https://www.eetimes.com/document.asp?doc_id=1324664

6LoWPAN


INTERNET
Router
Router
Three Network Architectures
Edge router
Backhaul link
Edge router
Edge router
R
H
Simple LoWPAN
Extended LoWPAN
Ad-Hoc LoWPAN

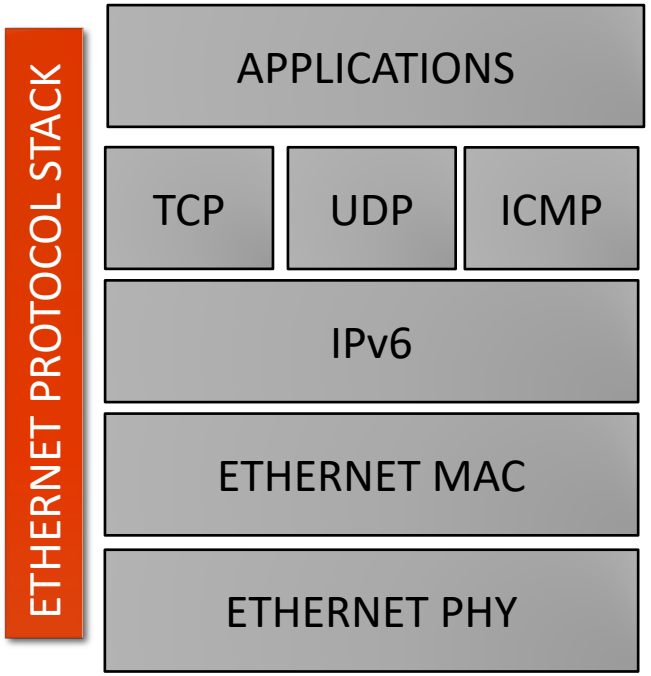
6LoWPAN

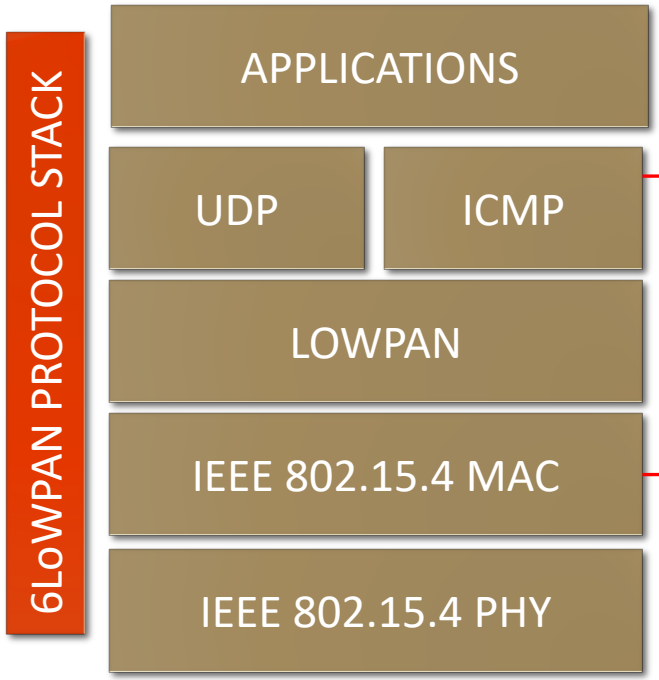


Three types of nodes:
Hosts → end-user sleepy device,
Routers → forward data inside the LoWPAN
Edge Routers → connect a LoWPAN to an external IPv6 network

6LoWPAN

❑ 6LoWPAN Protocol Stack vs Ethernet Protocol Stack





UDP is the most common TSP protocol with 6LOWPAN, since its header can be easily compressed …

6LoWPAN can work with other link-layer protocols beside IEEE 802.15.4.

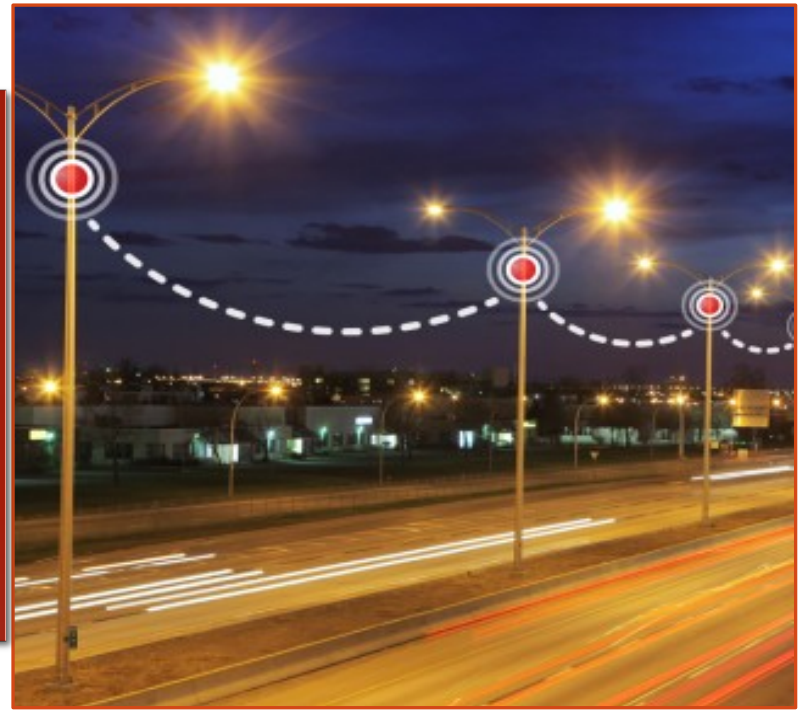
Requirements:

- ✧ Unique addressing
- ✧ Unicast transmissions
- ✧ MTU size > 30 bytes

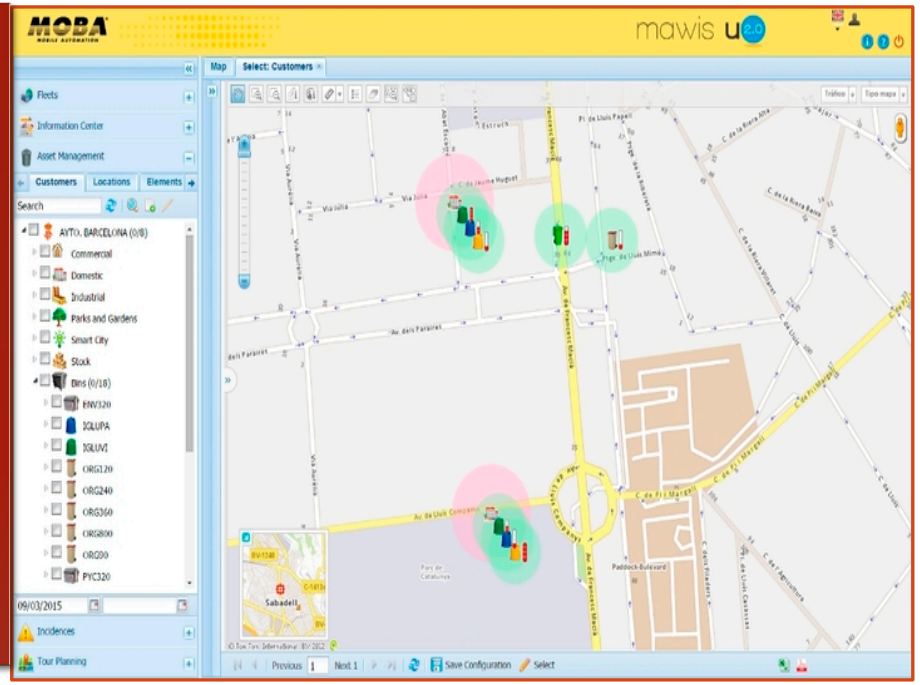
6LoWPAN

❑ Use-cases: Large-scale IoT Deployment

SMART LIGHTING SYSTEM



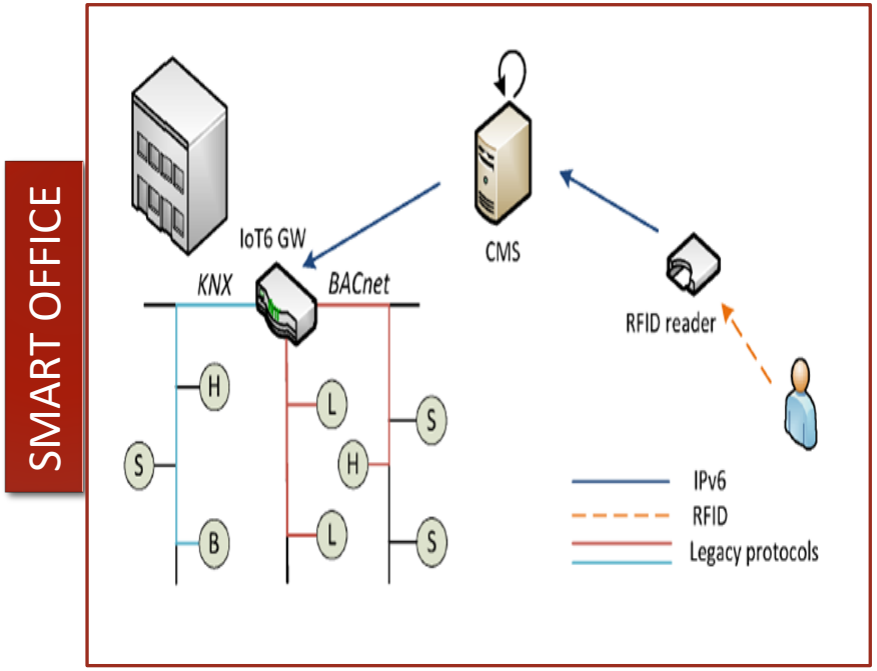
WASTE MANAGEMENT SYSTEM

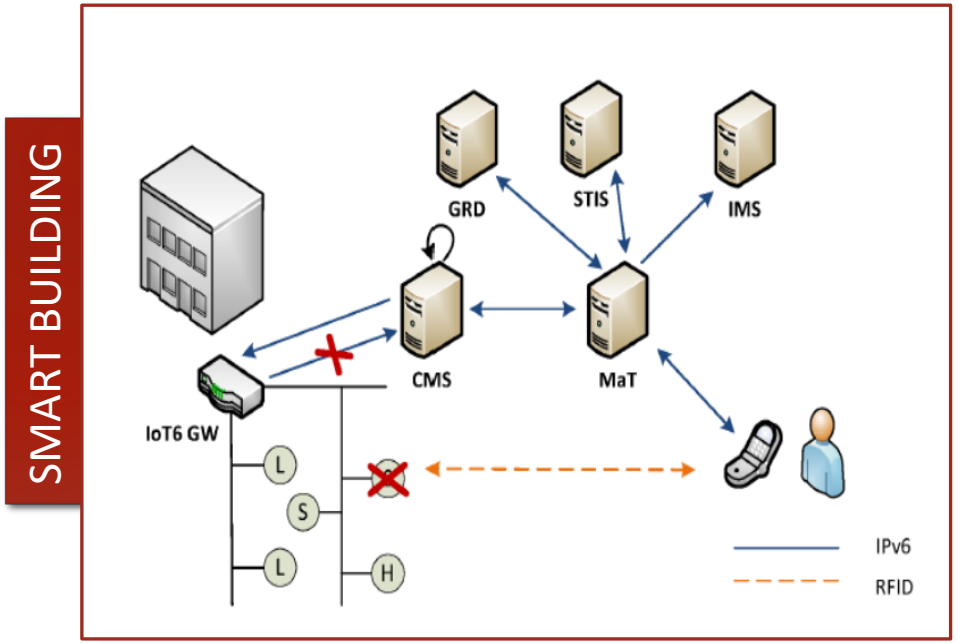


https://iot6.eu/iot6_%20use_cases

6LoWPAN

❑ Use-cases: Interoperable, Smart Environments

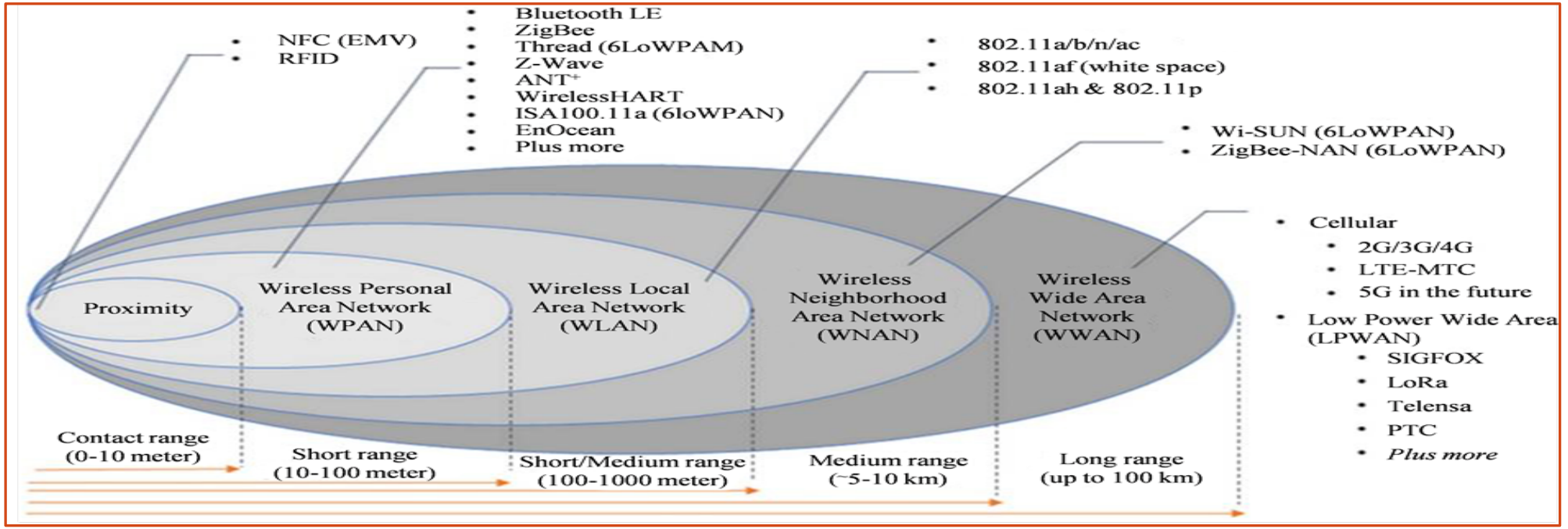




Digression: IEEE 802.15.4

- Low-power, low-cost technology for **Wireless Personal Area Networks** (WPANs)

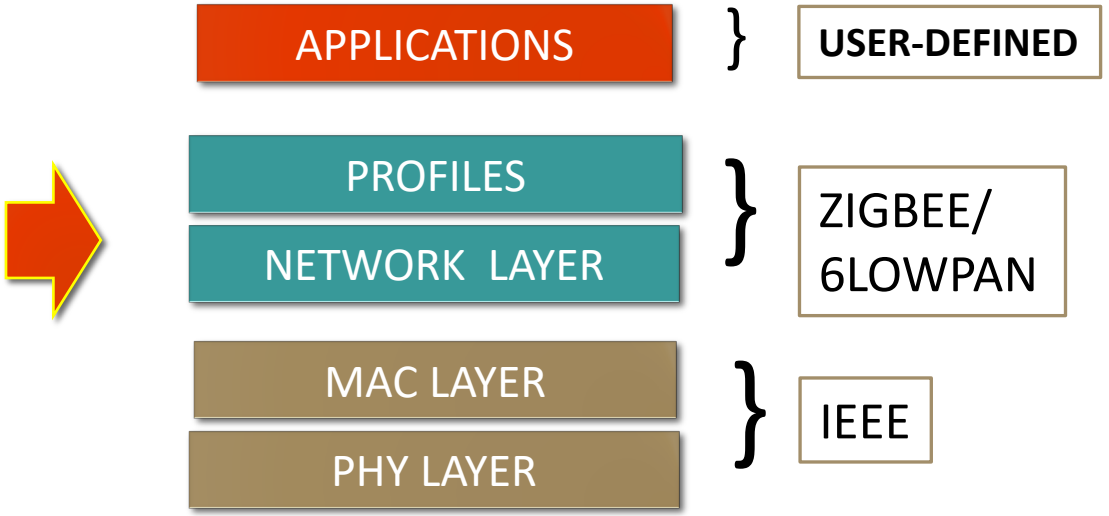
Source: http://file.scirp.org/Html/1-4000110_65802.htm



Digression: IEEE 802.15.4

❑ **IEEE 802.15.4** → standard for the deployment of WPAN. Characteristics: low complexity, low-power for low-datarate wireless connectivity among fixed and portable devices.

The specifications define the PHY techniques and MAC layer, while the upper layers are defined by other stacks (e.g. **Zigbee**).



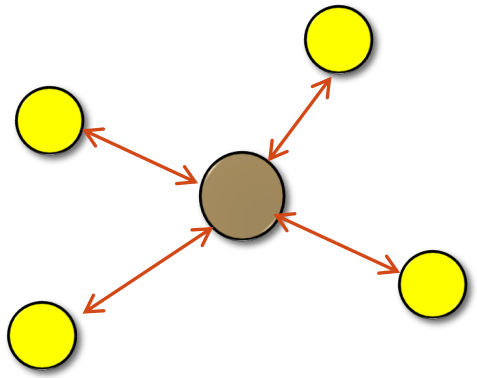
Digression: IEEE 802.15.4

❑ **IEEE 802.15.4** → standard for the deployment of WPAN. Characteristics: low complexity, low-power for low-datarate wireless connectivity among fixed and portable devices.

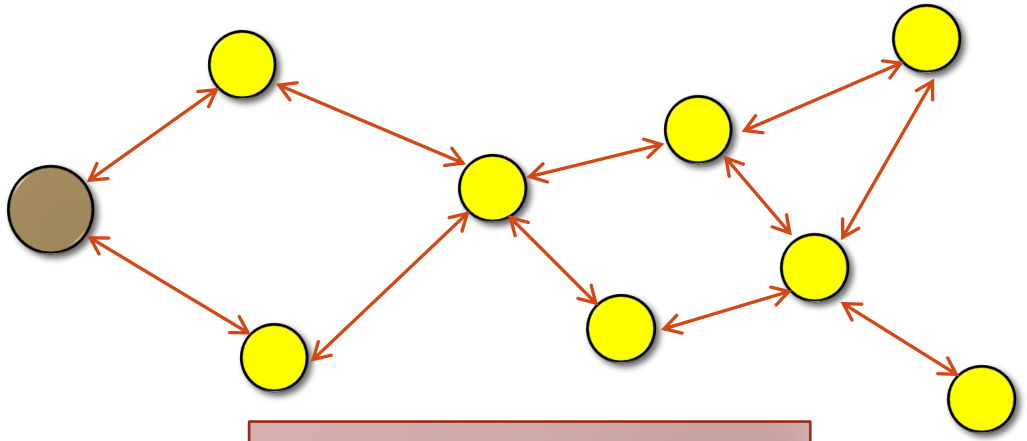
Feature	Description
Spectrum bands	2.4GHz, 915 MHz or 868 MHz
Data-rate	Up to 250 Kbs (2.4GHz)
Range	<30 meters
Channels	16 (2.4GHz)
Channel access	CSMA/CA or slotted CSMA/CA

Digression: IEEE 802.15.4

❑ **IEEE 802.15.4** → standard for the deployment of WPAN. Characteristics: low complexity, low-power for low-datarate wireless connectivity among fixed and portable devices.

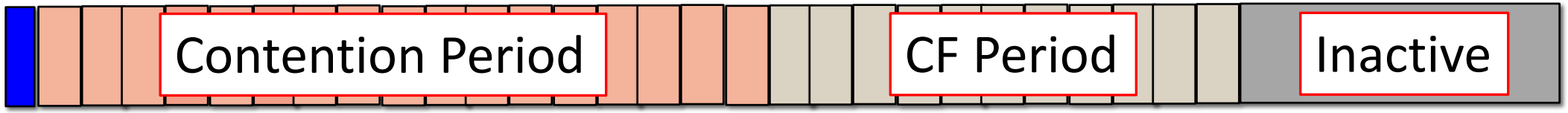


STAR TOPOLOGY



AD HOC TOPOLOGY

Digression: IEEE 802.15.4



Network **BEACON**, send by the PAN coordinator, and containing network-related info. Used also for synchronizing each device with the start of the contention-free operations.

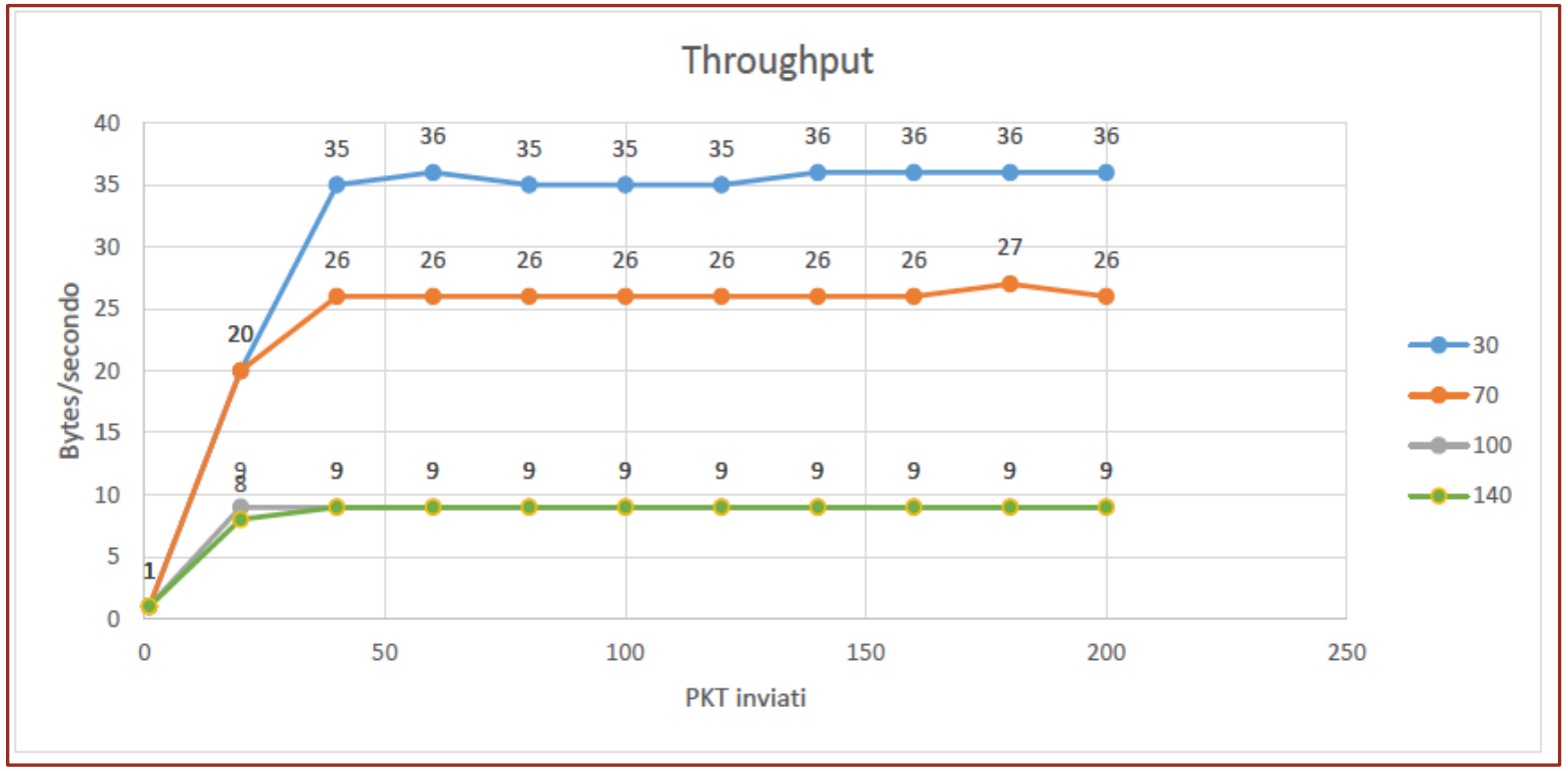
Contention-period slots. Accessed by using CSMA/CA protocol.

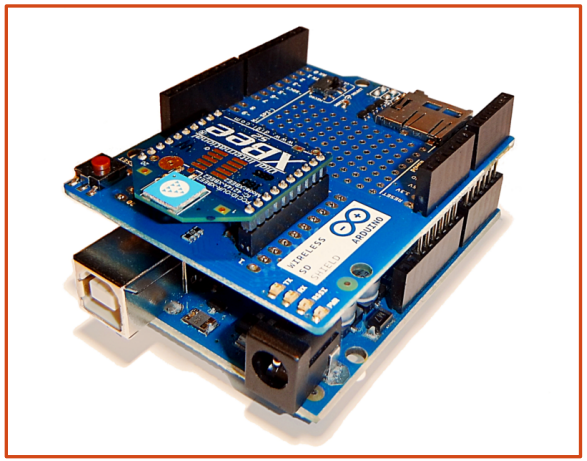
Contention-Free period slots. Reserved by PAN coordinator to applications with **QoS** requirements.

Inactive periods (needed for **energy saving** on battery-constrained devices)

Digression: IEEE 802.15.4

- Performance of **IEEE 802.15.4** networks (Arduino Xbee testbed).





Source: www.arduino.cc

6LoWPAN

- **Main operations**:
  - Device Addressing
  - Routing (different from forwarding)
  - Header Extensions
  - Header compression
  - Fragmentation
  - Bootstrapping & Device discovery
  - …

6LoWPAN: Addressing

- **IPv6 addresses** are typically formed **automatically** from the prefix of the LoWPAN edge router, and the MAC address of the wireless card.
- The IEEE 802.15.4 supports two MAC address format:
  - **64-bit EUI-64 address**

    `ACDE:4812:3456:7890` + `2001:0DB8:0BAD:FADE`

    EUI-64 MAC address Network Prefix
  - **48-bit EUI-64 address**

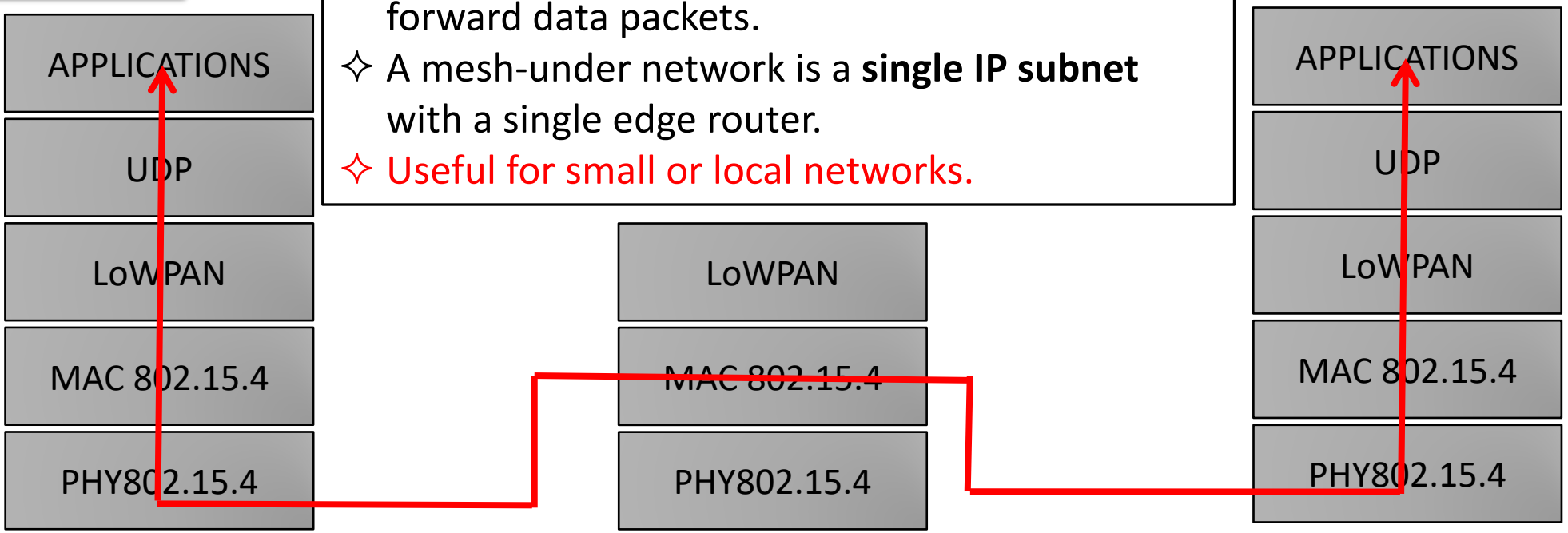
    PAN Network Identifier (16 bits) + 16 bits (zeros) + PAN Address (16 bits)

6LoWPAN: Routing

- 6LoWPAN supports **two different routing modes**

MESH-UNDER ROUTING

- Uses the layer-two (**MAC layer**) addresses to forward data packets.
- A mesh-under network is a **single IP subnet** with a single edge router.
- Useful for small or local networks.

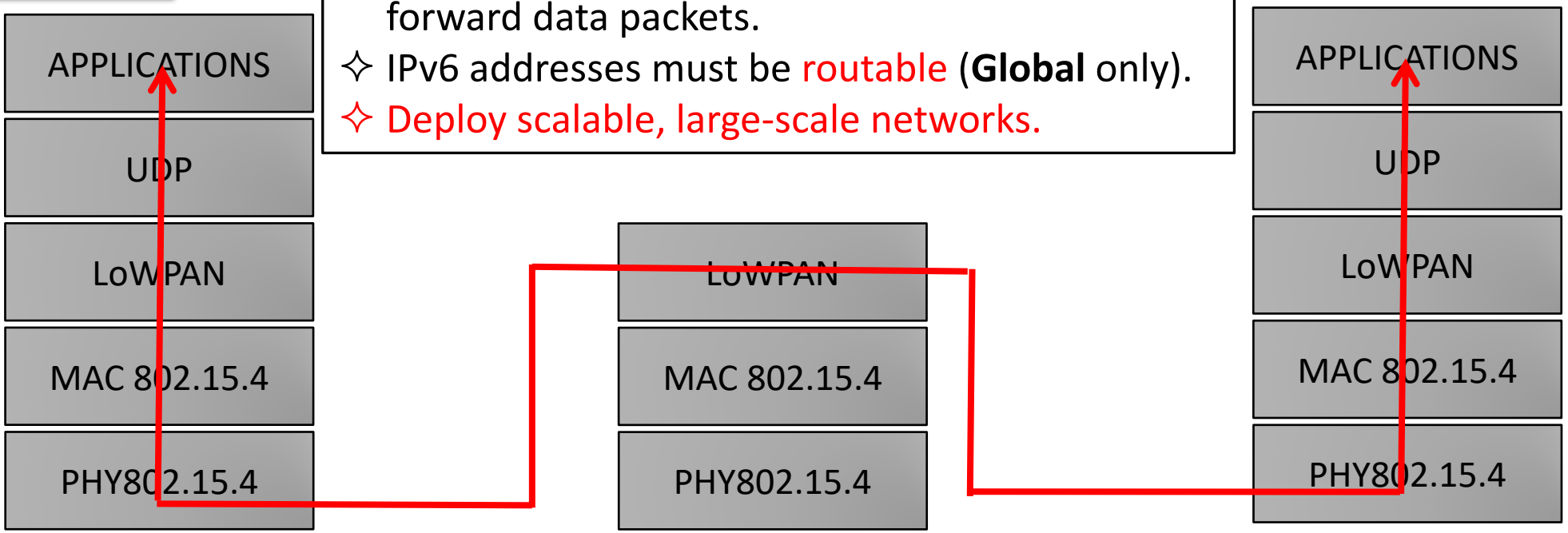


6LoWPAN: Routing

- 6LoWPAN supports **two different routing modes**

ROUTE OVER ROUTING

- Uses the layer-three (**IPv6**) addresses to forward data packets.
- IPv6 addresses must be routable (**Global** only).
- Deploy scalable, large-scale networks.



J. Olsson, 6LoWPAN Demystified, Whitepaper http://www.ti.com/lit/wp/swry013/swry013.pdf

6LoWPAN: Extension Headers

- Analogously to IPv6, 6LoWPAN uses the **Extension Headers** for the optional data and for specific use-cases.
- Two 6LoWPAN Extension Headers are defined:

FRAGMENT HEADER → used in case of packet fragmentation, see next slides

MESH HEADER → used by MESH_UNDER routing, it contains: <**ORIGINATOR_MAC**, **DESTINATION_MAC**, **NUM_HOPS_LEFT**>

6LoWPAN: Fragmentation

- All IPv6 subnetworks have to provide a **minimum MTU** of 1280 bytes (recommended: 1500 bytes).
  - IPV6 does provide its own fragmentation for datagrams larger than the minimum MTU (1280 bytes).
  - 6LoWPAN provides fragmentation in order to fit the size of 802.15.4 MTU (127 bytes)
  - **Mesh-Under** → fragments are reassembled at the destination. If any fragment is missing, **the complete packet must be re-transmitted by the source node**.

6LoWPAN: Fragmentation

- All IPv6 subnetworks have to provide a **minimum MTU** of 1280 bytes (recommended: 1500 bytes).
  - IPV6 does provide its own fragmentation for datagrams larger than the minimum MTU (1280 bytes).
  - 6LoWPAN provides fragmentation in order to fit the size of 802.15.4 MTU (127 bytes)
  - **Route-over** → fragments are reassembled at every hop (and fragmented again). If a fragment is missing, **the complete packet must be re-transmitted by the previous node**.

6LoWPAN: Fragmentation

- Fragment info are contained in the **Fragment Header**.
- All Fragments carry the same **tag value**, assigned sequentually by the source of fragmentation.

FIRST FRAGMENT

11000	SIZE	TAG

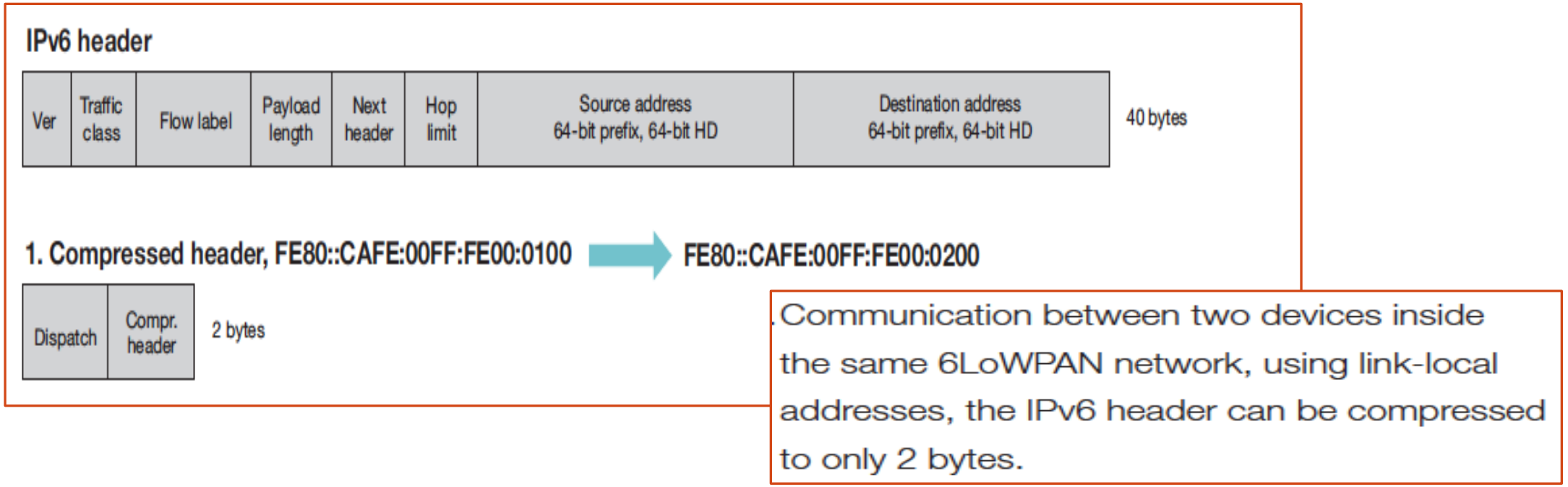
OTHER FRAGMENTs

11000	SIZE	TAG	OFFSET

J. Olsson, 6LoWPAN Demystified, Whitepaper http://www.ti.com/lit/wp/swry013/swry013.pdf

6LoWPAN: Header Compression

- 6LoWPAN can use **state-less** or **shared-context header** compression mechanisms.



J. Olsson, 6LoWPAN Demystified, Whitepaper http://www.ti.com/lit/wp/swry013/swry013.pdf

6LoWPAN: Header Compression

- 6LoWPAN can use **state-less** or **shared-context header** compression mechanisms.

IPv6 header

Ver	Traffic class	Flow label	Payload length	Next header	Hop limit	Source address 64-bit prefix, 64-bit HD	Destination address 64-bit prefix, 64-bit HD

40 bytes

2. Compressed header, 2001::DEC4:E3A1:FE24:9600 → **2001::4455:84C6:39BB:A2DD**

Dispatch	Compr. header	CID	Hop limit	Destination address 64-bit HD

12 bytes

Communication destined to a device outside of the 6LoWPAN network and the prefix for the external network is known, where the IPv6 header can be compressed to 12 bytes.

J. Olsson, 6LoWPAN Demystified, Whitepaper http://www.ti.com/lit/wp/swry013/swry013.pdf

6LoWPAN: Header Compression

- 6LoWPAN can use **state-less** or **shared-context header** compression mechanisms.

IPv6 header

Ver	Traffic class	Flow label	Payload length	Next header	Hop limit	Source address 64-bit prefix, 64-bit HD	Destination address 64-bit prefix, 64-bit HD

40 bytes

3. Compressed header, 2001::DEC4:E3A1:FE24:9600 → **2001::4455:84C6:39BB:A2DD**

Dispatch	Compr. header	CID	Hop limit	Source address 64-bit prefix	Destination address 64-bit prefix, 64-bit HD

20 bytes

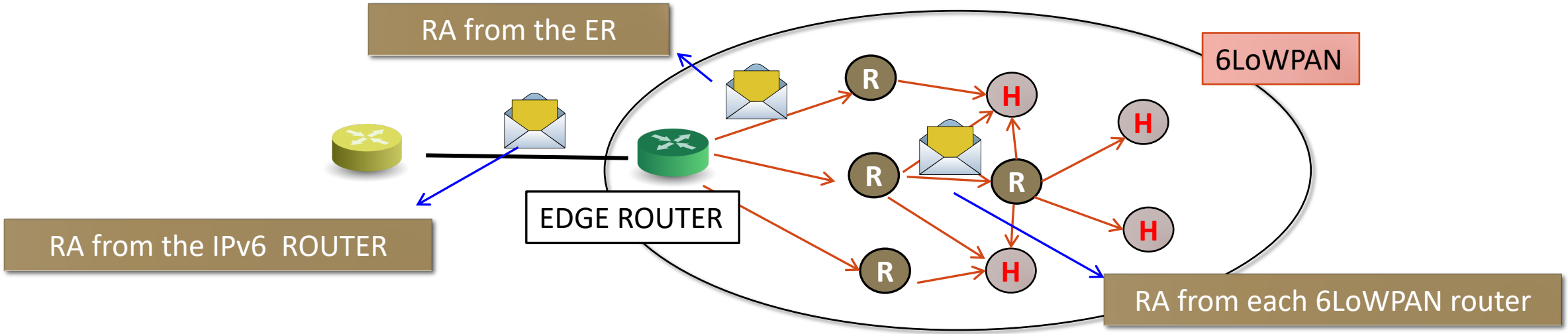
Similar to 2, but without knowing the prefix of the external device, that gives an IPv6 header of 20 bytes.

6LoWPAN: Device Discovery

- The **IPv6 Neighbour Discovery Protocol** is used by IPv6 nodes to find routers, to determine their link-layer address and to maintain reachibility info about the paths.
  - Routers send **Announcement messages** (RA) in multicast, attaching their network prefix.
  - IPv6 nodes can solicit a RA message by using a **Router Solicitation** (RS) message.
  - Each IPv6 node builds its own address: <Prefix, MAC>

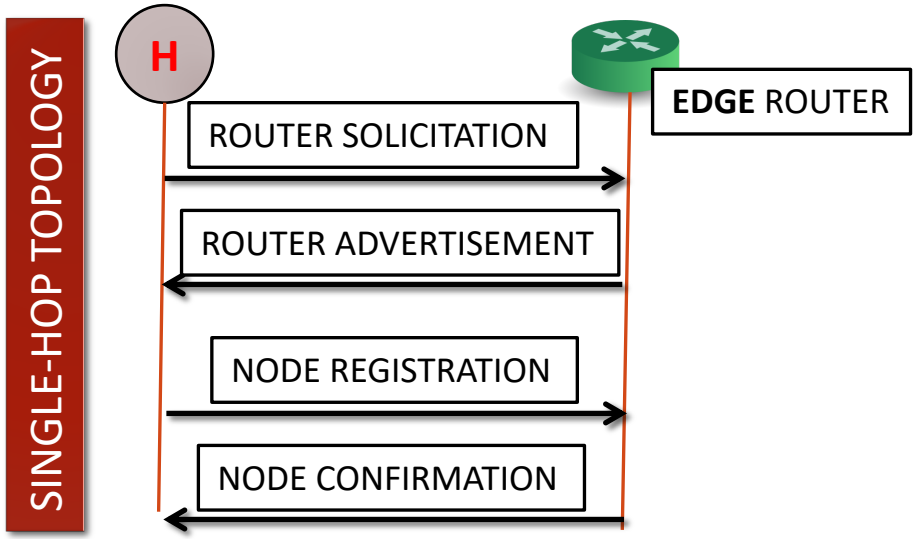
6LoWPAN: Device Discovery

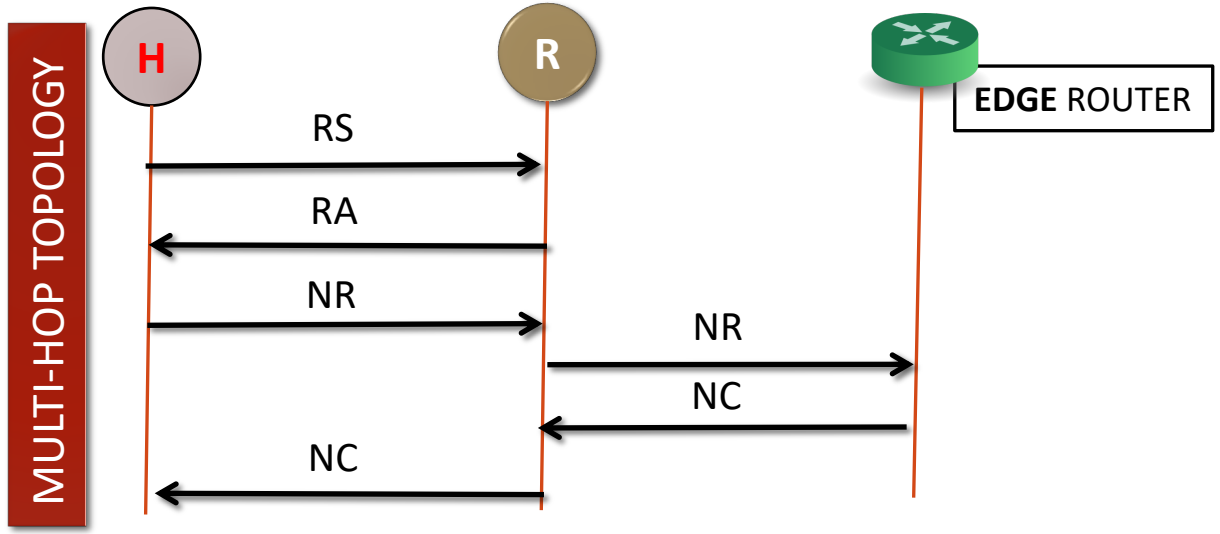
- Differences compared to the standard NDPv6 protocol
  - In 802.15.4 networks, 6LoWPAN nodes might belong to different broadcast domains (e.g. multi-hop scenarios).
  - RA messages must be **flooded** in the entire 6LoWPAN.



6LoWPAN: Device Discovery

- Differences compared to the standard NDPv6 protocol.
  - The 6LoWPAN Edge Router maintains a **whiteboard** of all the IPv6 address registered in the 6LoWPAN.
  - It also performs **Duplicate Address Detection** (DAD).





RPL IETF specifications (RFC 6550) -- https://tools.ietf.org/html/rfc6550

RPL Protocol: Routing over 6LoWPAN

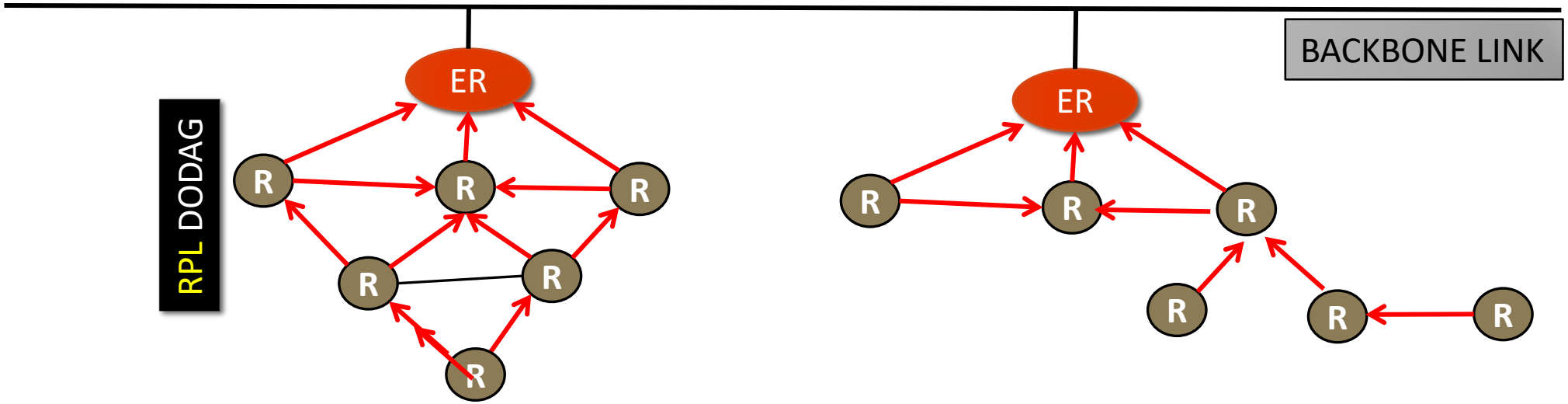
- **RPL** → IPv6 Routing Protocol for Low-Power and Lossy Networks
  - ✧ Standardized by the IETF in 2011 (current draft: RFC 6550)
  - ✧ De Facto **standard routing protocol** for IoT scenarios characterized by the presence of low-power, resource-constrained devices.
  - ✧ It supports: point-to-point, point-to-multipoint and multipoint-to-point communications.
  - ✧ It separates **packet processing and forwarding** from the routing optimization objective (e.g. min energy, maxthroughput, min delay, etc).
  - ✧ It can be used to disseminate IPv6 or 6LoWPAN specific info (e.g. neighbour discovery).
  - ✧ It **does not rely on any specific link-layer protocol** (although it is commonly coupled with the IEEE 802.15.4 standard).

RPL Protocol: Routing over 6LoWPAN

- RPL creates a routing topology in the form of a **Destination-Oriented Directed Acyclic Graph** (DODAG)
  - Directed graph without cycles, oriented towards a root node (the edge router).

RPL Protocol: Routing over 6LoWPAN

- In case of Extended LoWPANs (i.e. presence of multiple Edge Routers), RPL might create **multiple disjoint DODAGs**, routed at different ER.



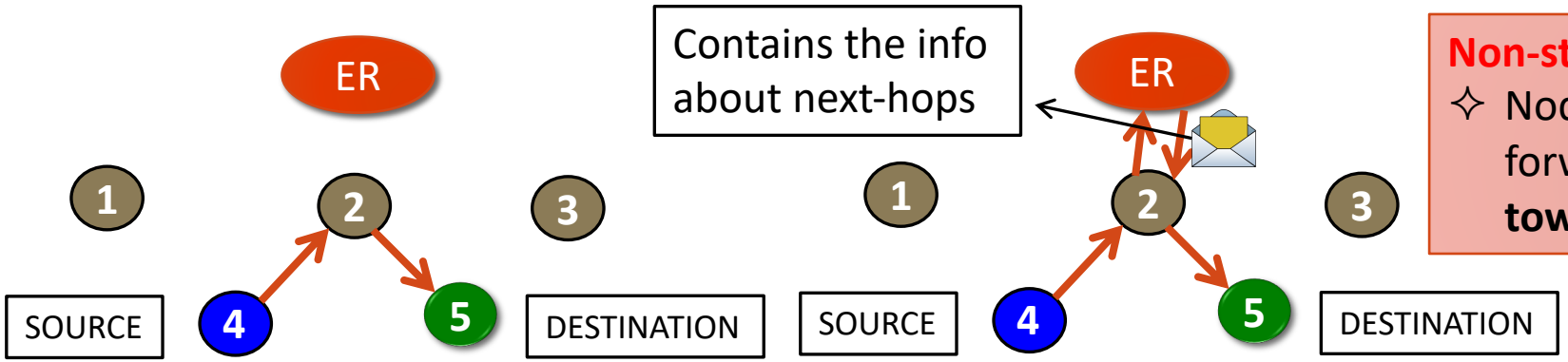
RPL Protocol: Routing over 6LoWPAN

- In order to create and maintain the DODAG, the RPL protocol introduces the **following control packets**:
  - **DIO** (DODAG Information Object) → used to enstablish the upward path (from leafs to root)
  - **DAO** (Destination Advertisment Object) → used to enstablish the downlink path (from root to leafs)
  - **DIS** (DODAG Information Solicitation) → used by an internal node in order to solicitate the transmission of DIO messages
  - **DAO-ACK** (Destination Advertisement Object Acknowledgement)

RPL Protocol: Routing over 6LoWPAN

- Two modes of operation: **storing** and **non-storing**
  - **Storing** → each node keeps a **routing entry** for all the destinations reachable via its sub-DODAG.
  - **Non-Storing** → the root is the only network node maintaining routing information; source routing is used for downward routing.

Storing Mode:
- Node 4 forwards data toward Node 2
- Node 2 **stores routing info for all its subgraph** (nodes 4 and 5)



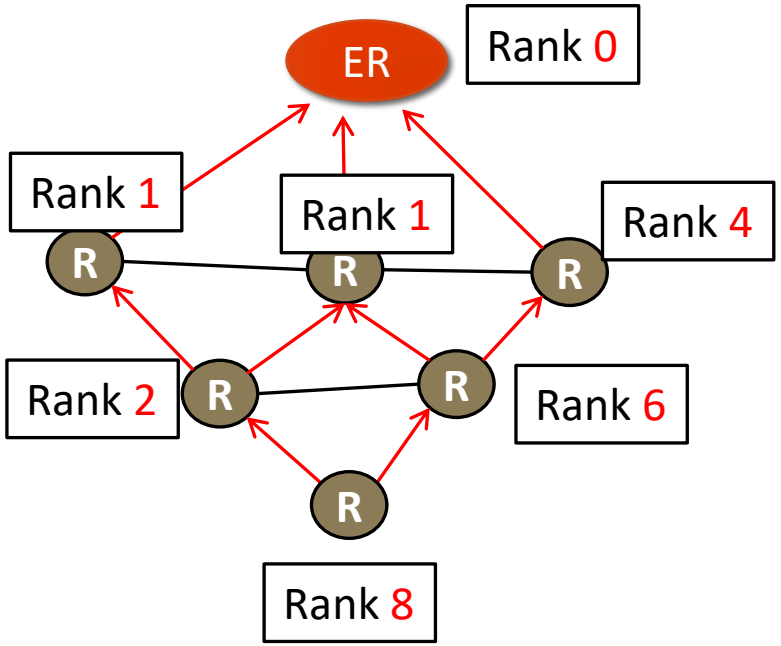
Non-storing Mode:
- Node 4 always forwards data **toward the root**

RPL Protocol: Routing over 6LoWPAN

- Each node of the DODAG has its own **rank** value.

PROPERTIES

- **Abstract numeric value**, expression of a relative position within a DODAG Version.
- Rank of the nodes must monotonically decrease towards the DODAG destination.
- Rank is used to avoid and detect loops.



HOW TO COMPUTE IT?

- Rank is computed according to the **Objective Function** in use (see next slides)

RPL Protocol: Routing over 6LoWPAN

- Creation of the **upward paths** (assumed at start-up)

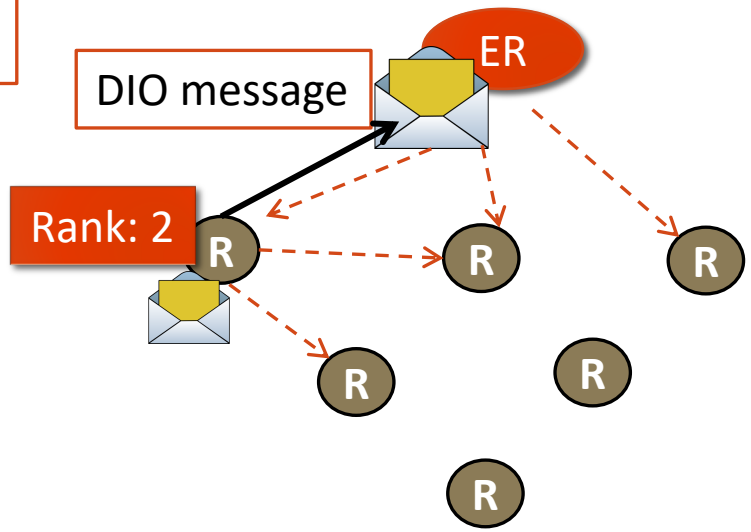
1. The Edge router creates the **DIO** message, containing its rank and DODAG id, and sends it in **multicast.**

RECEIVING **NODES**

2. Each node establishes the upward link toward **the sender.**

3. Each node computes its own rank value, based on the **root's rank and on the Objective Function.**

4. Each node rebroadcasts the DIO message (following the **Trickle** algorithm), by including its own computed rank**.**



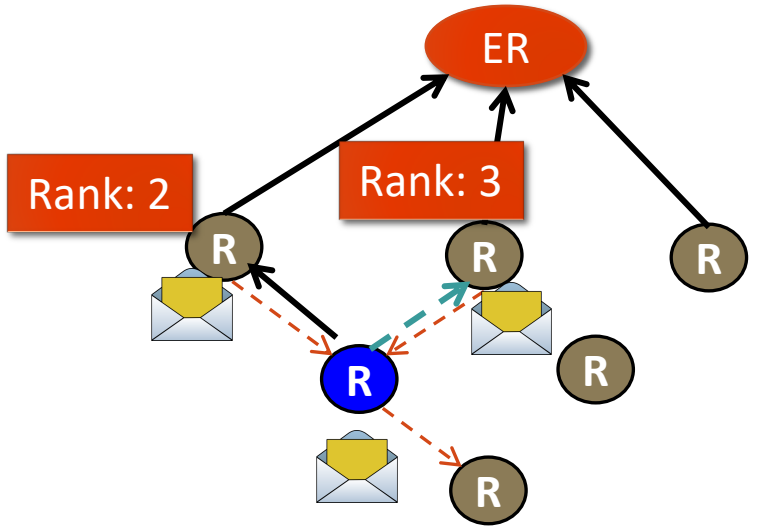
RPL Protocol: Routing over 6LoWPAN

- Creation of the **upward paths** (assumed at start-up)

A node receiving multiple DIO messages (e.g the blue node)

2. Based on the used metric and constraints defined by the Objective Function, it chooses an appropriate parent:
   - Multiple parents can be established, but a **preferred parent** is selected;
   - If the node has already its own rank, and the received one is greater than the local rank, the DIO message is discarded (**loop avoidance**)

3. As before, each node rebroadcasts the DIO message (following the **Trickle** algorithm), by including its own computed rank**.**



The routing procedure ends when reaching the leaf nodes.

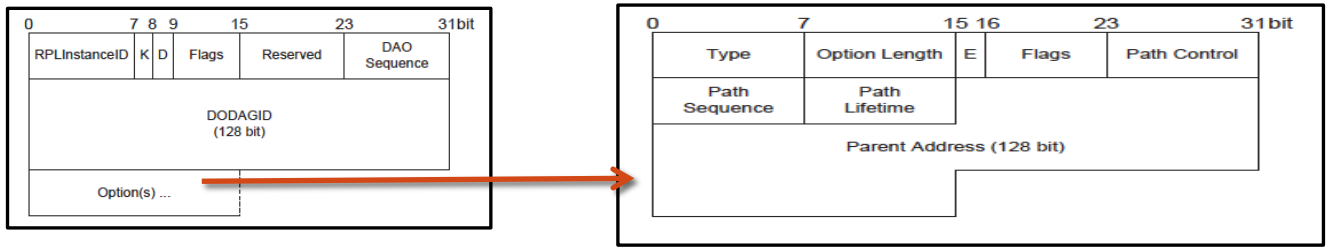
RPL Protocol: Routing over 6LoWPAN

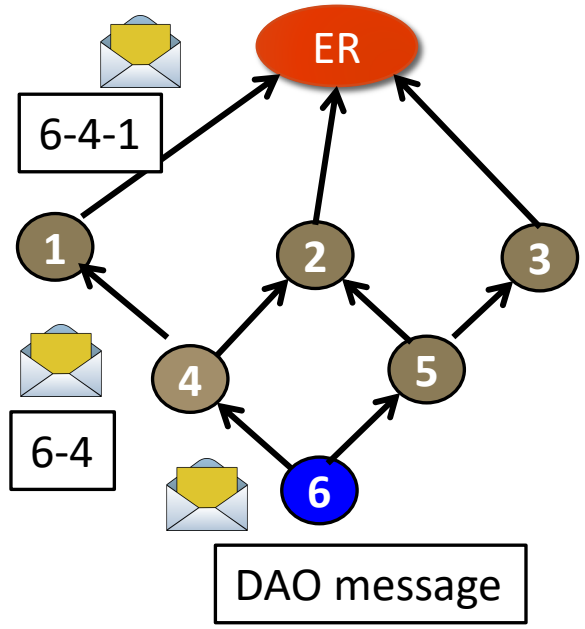
- Creation of the **downward paths** (from leaf to edge router)

NON-STORING MODE

1. Each node periodically generates a DAO message and sends it to the destination, by using the upward path established through the DIO message.

2. All the intermediate parents extend the DAO message by adding their IPv6 address in the **Transit Information Option.**





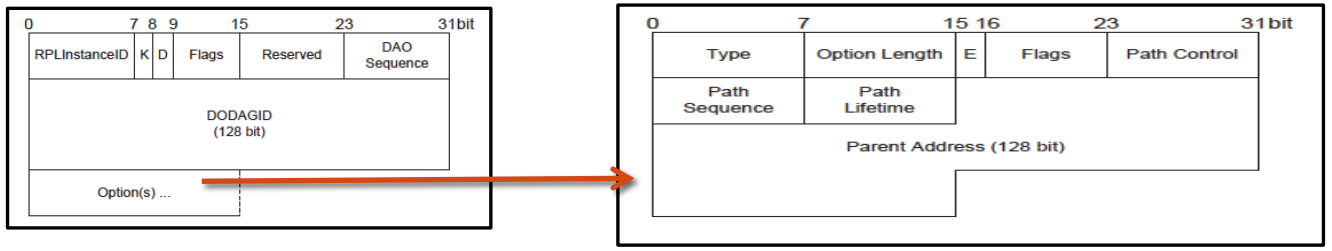
RPL Protocol: Routing over 6LoWPAN

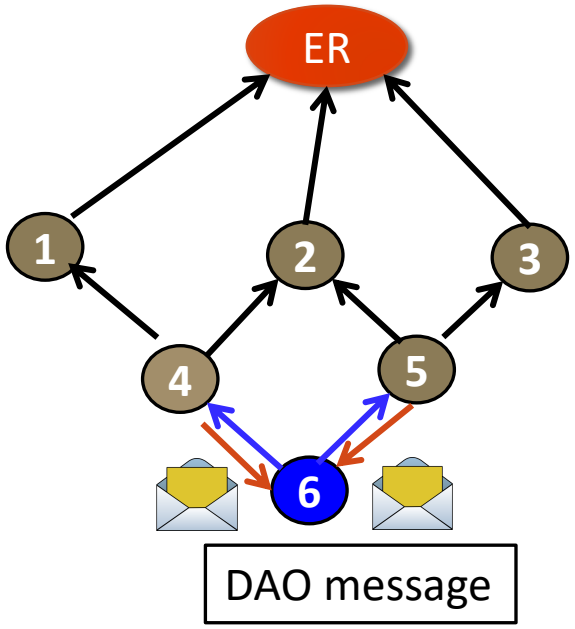
- Creation of the **downward paths** (from leaf to edge router)

STORING MODE

1. Each node periodically generates a DAO message and sends it to all parents node (differently to the previous case, the message is not forwarded toward the root).

2. Each parent maintains additional routing tables for all the nodes of its sub-DODAG.





Trickle algorithm – IETF specifications: https://tools.ietf.org/html/rfc6206

RPL Protocol: Routing over 6LoWPAN

- **Trickle** algorithm → data dissemination scheme for **lossy shared medium** (e.g. low-power and lossy networks).
  - ✧ It can be applied to a wide range of protocol design problems (beside our topic, i.e. the DIO message dissemination in RPL)
  - ✧ Three **configuration parameters**: the minimum interval size I_{min}, the maximum interval size I_{max}, and a redundancy constant k.
  - ✧ In addition, Trickle maintains **three variables**:
    - ✓ I → the current interval size.
    - ✓ t → a time within the current interval.
    - ✓ c → a counter.

Trickle algorithm – IETF specifications: https://tools.ietf.org/html/rfc6206

RPL Protocol: Routing over 6LoWPAN

- The **Trickle** execution follows five rules:
  1. At startup, it sets I to a value in the range of [Imin, Imax], c to 0 and t to a random point in the interval, [I/2, I];
  2. Whenever Trickle hears a transmission that is "**consistent**", it increments the counter c;
  3. At time t, Trickle transmits if and only if the counter c is less than the redundancy constant k.
  4. When the interval I expires, Trickle doubles the interval length (I).
  5. If Trickle hears a transmission that is "**inconsistent**" and I is greater than I_{min}, sets I to I_{min} and t to a random point in the interval [I/2, I] (step 1).

The meaning of consistent and inconsistent depends on the specific use-case!

Trickle algorithm – IETF specifications: https://tools.ietf.org/html/rfc6206

RPL Protocol: Routing over 6LoWPAN

- The **Trickle** execution follows five rules:
  1. At startup, it sets I to a value in the range of [Imin, Imax], c to 0 and t to a random point in the interval, [I/2, I];
  2. Whenever Trickle hears a transmission that is "**consistent**", it increments the counter c;
  3. At time t, Trickle transmits if and only if the counter c is less than the redundancy constant k.
  4. When the interval I expires, Trickle doubles the interval length (I).
  5. If Trickle hears a transmission that is "**inconsistent**" and I is greater than I_{min}, sets I to I_{min} and t to a random point in the interval [I/2, I] (step 1).

EXAMPLE: CONSISTENCY of TOPOLOGY in RPL-DIO messages …

RPL Protocol: Routing over 6LoWPAN

- The **Objective Function** (OF) defines the specific metrics/constraints to use for finding minimum cost paths.
  - How to compute the rank;
  - How to select the parents (and the preferred parent);
  - How to compute the path cost.
    - **EXAMPLE1**. Determine the shortest route (METRIC) by avoiding low-energy nodes (CONSTRAINT).
    - **EXAMPLE2**. Determine the lowest end-to-end delay (METRIC) by avoiding low-quality links (CONSTRAINT).

Minimum Rank with Hysteresis Objective Function– IETF specifications: https://tools.ietf.org/html/rfc6719

RPL Protocol: Routing over 6LoWPAN

- Two objective functions have been defined so far:
  - **OF0**: Objective Function Zero →
    use hop count as default routing metric.
  - **OF1**: Minimum Rank with Hysteresis Objective Function →
    Select routes which minimize an additive metric.
    Default Metric: Expected Transmission Number (ETX)

$$PRR(\rho) = \frac{Number\ of\ received\ packets}{Number\ of\ sent\ packets}$$

$$ETX = \frac{1}{PRR_{down} \cdot PRR_{up}}$$