IoTSSC - Addressing and Routing in IoT

Image: TRENDR

How to enable end-to-end connectivity?

- You will need some form of global addressing (unique identifiers)
- A mechanism to transfer information between different end points (routing protocol)
- While dealing with IoT specific constraints (reduced computation capabilities, small messages, limited energy)

The need for IPv6

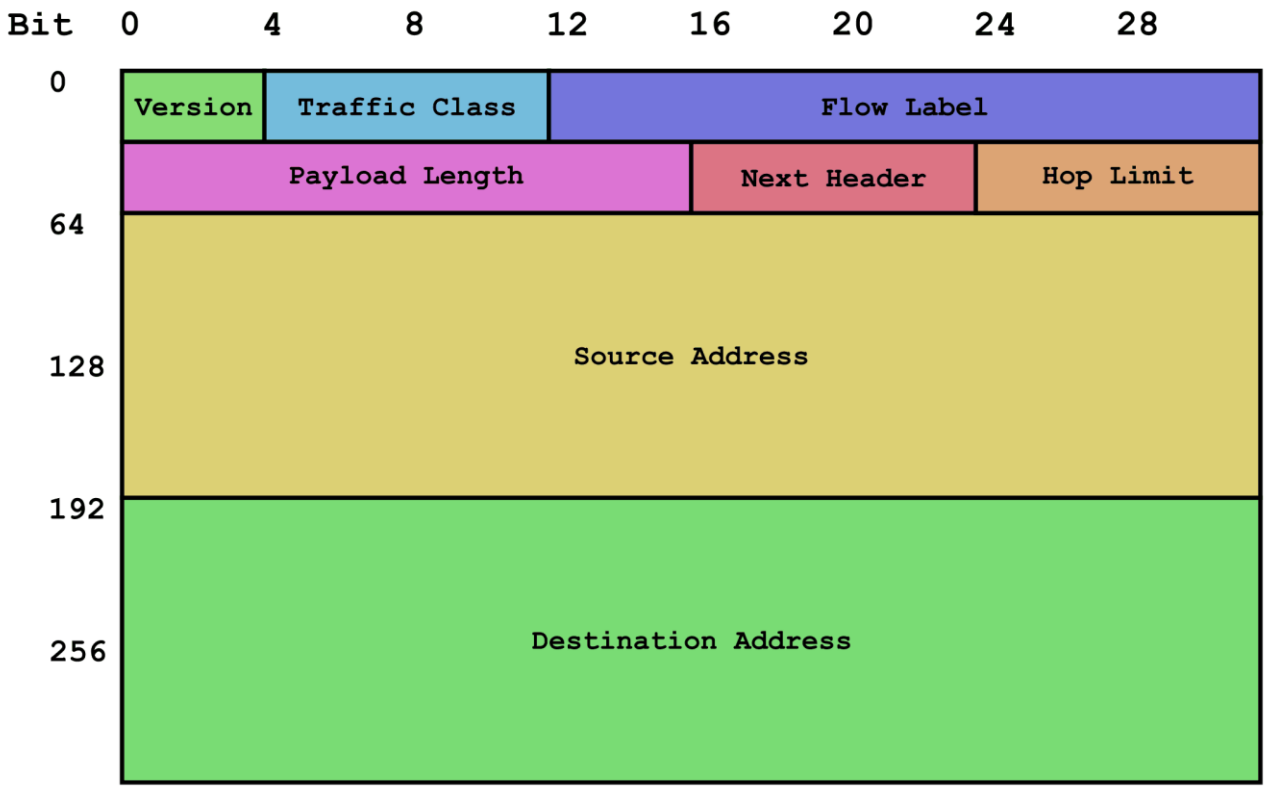
- You have seen that IPv4 has underpinned the growth of the Internet and does solve the device addressing issue.
- However, IPv4 is running out of addresses, especially in the context of trillions of IoT devices expected to be rolled out in the future.
- You may know that NAT permits sharing a single *public* IP address among multiple hosts, by assigning those *private* addresses; however, NAT suffers though from serious problems, e.g. breaks up layered designs.
- IPv6 is the only long-term solution: move to larger addresses – 128 bits (addressing 2^{128} UNIQUE interfaces).

IPv6 development

- Work on specification began in 1990. Currently specified by RFC 2460 to RFC 2466
- Some of the major goals:
    1. Support huge number of hosts
    2. Reduce the size of routing tables
    3. Simplify protocol → allow for faster packet processing
    4. Improve security
    5. Allow host roaming without address changing
- In general, IPv6 is not compatible with IPv4, but is compatible with Internet control and transport protocols such as ICMP, OSPF, BGP, TCP, UDP, etc.

IPv6 header

- Header much simplified as compared to IPv6 (7 fields vs. 13) and has better support for options.



IPv6 header (fixed part)

Fixed part of the header: 40 Bytes.

Version: 0110 (6) – Let routers know about packet type

Traffic Class: Used to distinguish different classes of services – useful for real-time traffic with strict req.

Flow Label: Marks groups of packets that should be treated in the same way, sort of connection oriented flavour

Payload Length: Similar to 'Total Length' in IPv4, but header length omitted here.

Next Header: Points to the first optional extension header (if any). The last header uses this field to specify the transport layer protocol, e.g. TCP, UDP)

Hop Limit: Same functionality as TTL in IPv4.

IPv6 addressing

- New notation uses eight groups of hexadecimal digits separated with colons.

- Example:

  `8000:0000:0000:0000:0123:4567:89AB:CDEF`

IPv6 addressing

To reduce notation 3 optimisations are authorised:

1. Leading zeros within a group can be omitted

2. One or more groups of 16 zero bits can be replaced by a pair of colons

   `8000::123:4567:89AB:CDEF`

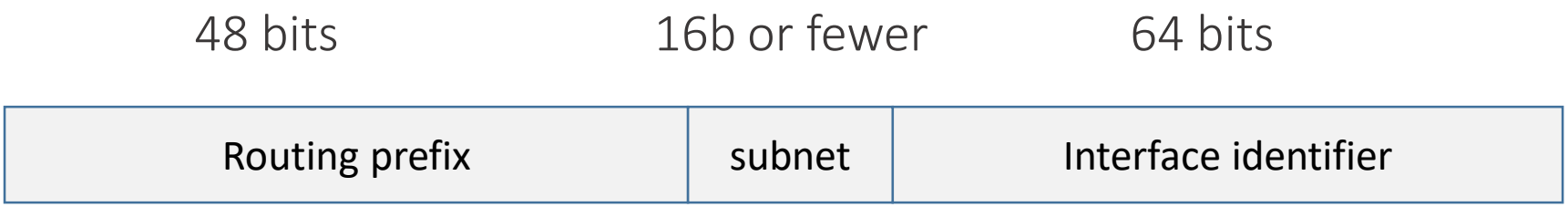
3. IPv4 addresses can be written as a pair of colons followed by decimal representations

   `::192.31.20.46`

Address types

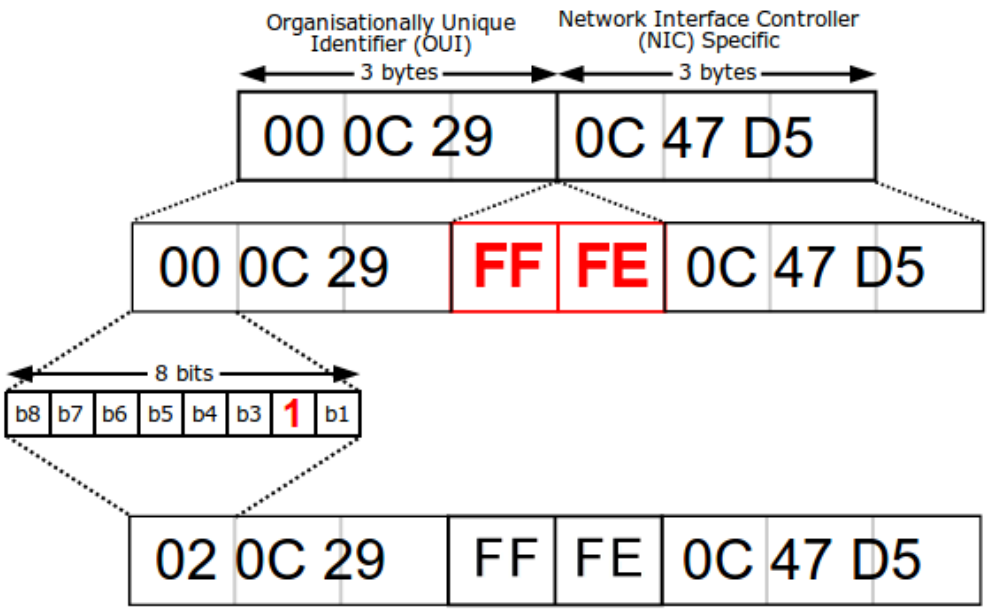
Prefix	Description	IPv4 equivalent
::/128	**Unspecified** (used at boot up)	0.0.0.0
::1/127	**Loopback**	127.0.0.1
::ffff/96 Example: ::ffff:192.0.2.47	**IPv4 mapped** (used to embed IPv4 addresses into IPv6)	No equivalence
fc00::/7 Example: fdf8:f53b:82e4::53	**Unique Local Addresses (ULAs)** Reserved for local use and are not public. (might not be unique)	Private addresses: 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
fe80::/10 Example: fe80::200:5aee:feaa:20a2	**Link-Local Addresses** Used on a single link or a non-routed common access, e.g. Eth. LAN. Not necessarily unique outside Link.	169.254.0.0/16
2000::/3	**Global Unicast**	No equivalent single block

Unicast addresses



- The network prefix (the routing prefix combined with the subnet id) is contained in the most significant 64 bits of the address.
- The size of the routing prefix may vary; a larger prefix size means a smaller subnet id size.
- The bits of the subnet id field are available to the network administrator to define subnets within the given network.
- The 64-bit interface identifier is either automatically generated from the interface's MAC address using the modified EUI-64 format, obtained from a DHCPv6 server, automatically established randomly, or assigned manually.

Modified EUI-64



- MAC address:
  `00:0C:29:0C:47:D5`
- Network prefix:
  `2001:db8:1:2::/64`

- Resulting host address:
  `2001:db8:1:2:020C:29ff:fe0c:47d5`

Extension headers

6 extensions defined for extra functionality

- Routing – Extended routing, e.g.IPv4 loose source route
- Fragmentation – Fragmentation and reassembly
- Authentication – Integrity and authentication, and security
- Encapsulating Security Payload – Confidentiality
- Hop-by-Hop options – Special options that require hop-by-hop processing
- Destination options – Optional information to be examined by the destination node

IPv6 over IEEE 802.15.4 (6LoWPAN)

Challenges of E2E IoT Networking

Protocols such as IEEE 802.15.4 have limited packet sizes (standard size is 127 bytes)

- IPv6 fixed header: 40B; UDP header: 8B
- 802.15.4 header: 25B
- Security options may add: 21B

Only 33B left for data! Some compression required.

- IPv6 requires MTU=1280B

Packet fragmentation and reassembly is required.

IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN)

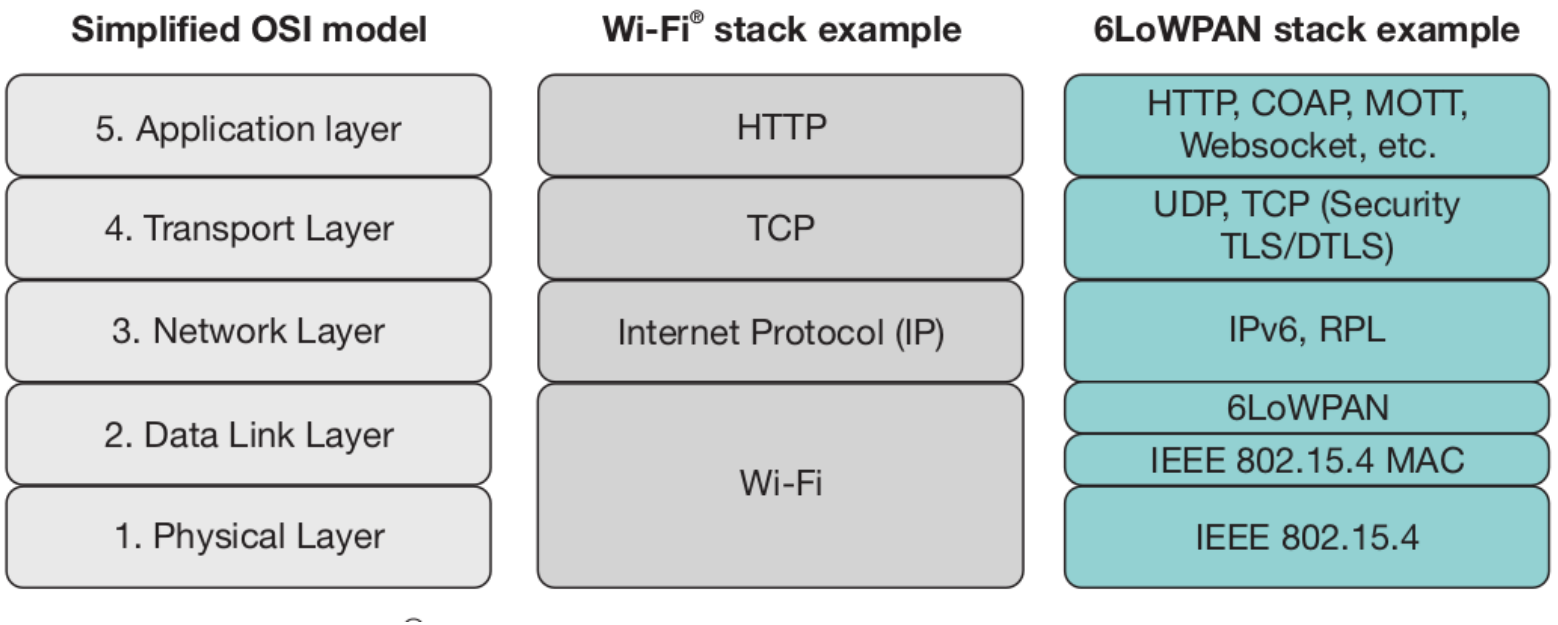
- 6LoWPAN (RFC 4944) introduces an adaptation layer to enable the transport of IPv6 packets over 802.15.4 links

- Main functions:
  - Fragmentation / reassembly
  - Compression of IPv6 and UDP headers

- Why not use ZigBee (also over 802.15.4)? -> ZigBee cannot easily communicate with other protocols (but more energy efficient)

6LoWPAN stack

*Texas Instruments: 6LoWPAN demystified



- CoAP (Constrained Application Protocol), MQTT (Message Queue Telemetry Transport) - like HTTP but IoT focused (resource discovery, publish/subscribe, etc.) – upcoming lecture
- RPL (Routing Protocol for Low-Power and Lossy Networks) - today

Header compression

Key idea: omit fields if can be derived from the link layer / context

Three scenarios:

1. Communication between devices on the same network – compress header to two bytes
2. Communication with a device outside local network, network prefix known – compress to 12 bytes
3. Communication with device on external network, device prefix not known – compress to 20 bytes (50%)

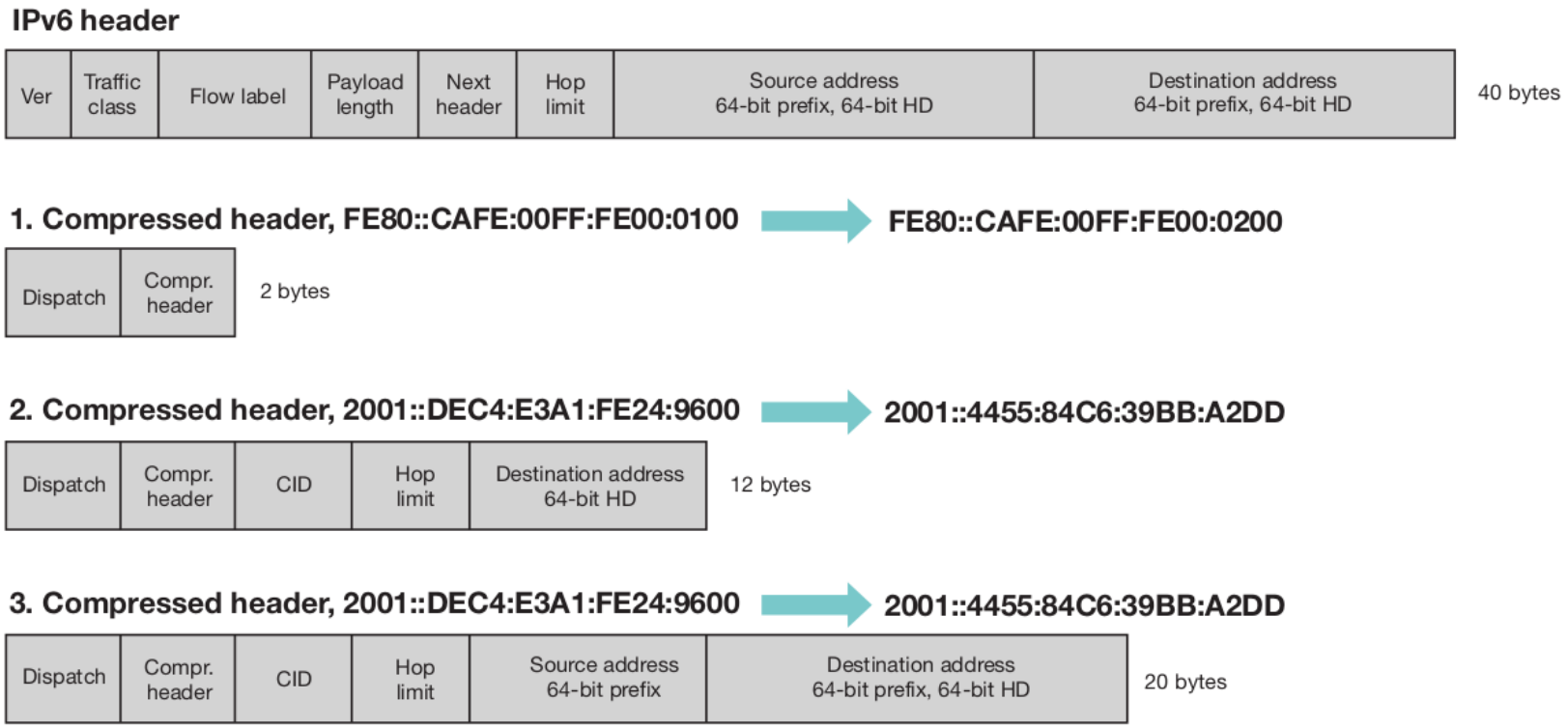
Header compression

All packets prefixed with a 1-byte a dispatch code (encapsulation header)

Pattern	Header type
00 XXXXXX	NALP - Not A LoWPAN Packet
01 000001	IPv6 - Uncompressed IPv6 addresses
01 000010	LOWPAN_HC1 – Compressed IPv6 header
01 111111	ESC - Additional Dispatch octet follows
…	Others reserved + broadcast, fragmentation, mesh

First fragment’s header includes the datagram size (11 bits) and a datagram tag (16 bits).

Example



*Texas Instruments: 6LoWPAN demystified

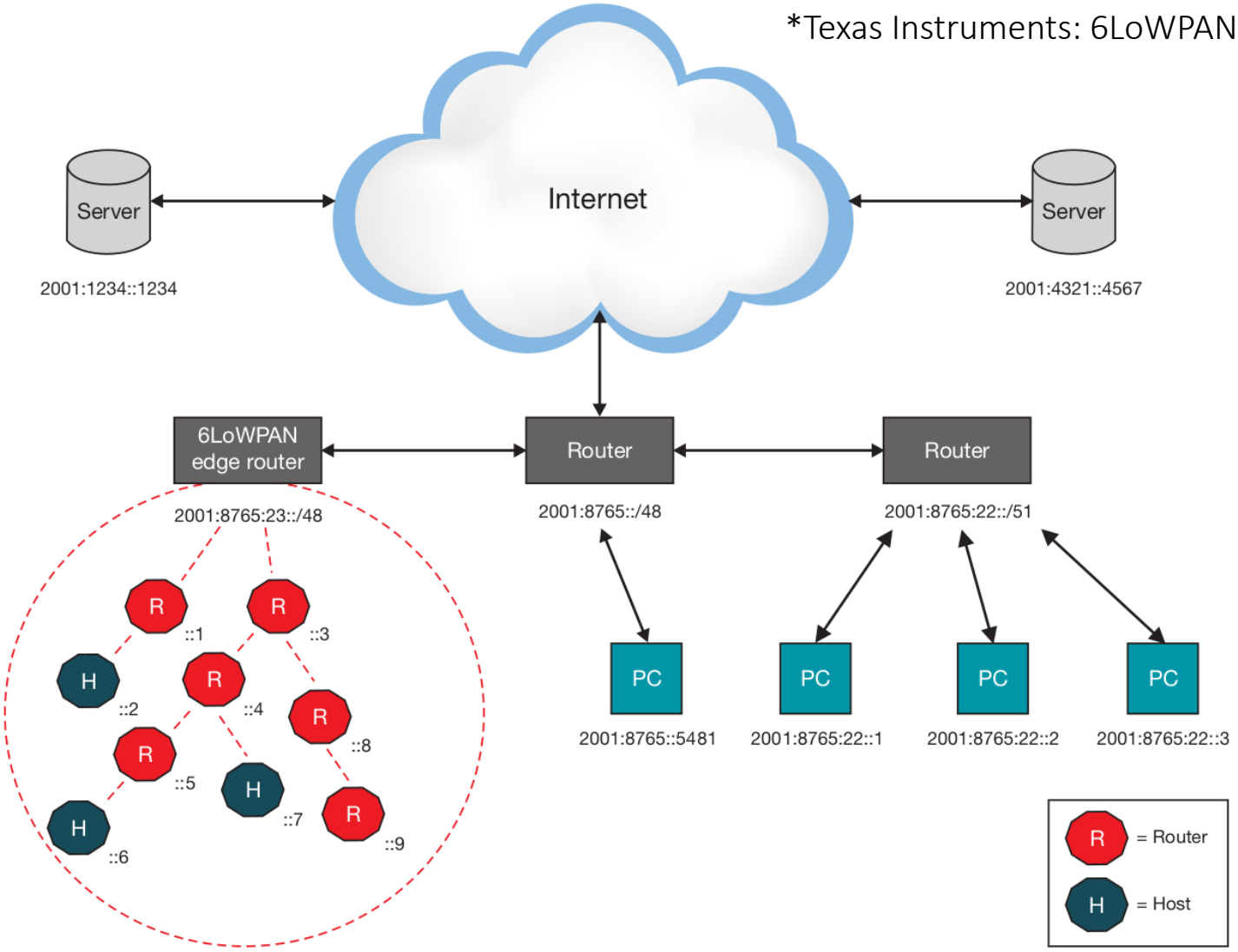
Auto-configuration

- Devices assign themselves addresses without the need for a DHCP server
- Generates link-local unicast address (FE80::IID)
- IID – based on IEEE 802.15.4 EUI-64 address, 16-bit short address, or both.
- Router Solicitation (RS) used to discover network prefix – this can be omitted in local communication
- Receive Router Advertisement (RA) – network prefix
- Send a neighbour solicitation (NS) message to check if address in use – Duplicate Address Detection (DAD)

IPv6 Routing Protocol over Low-power and Lossy Networks (RPL)

Different node functionality

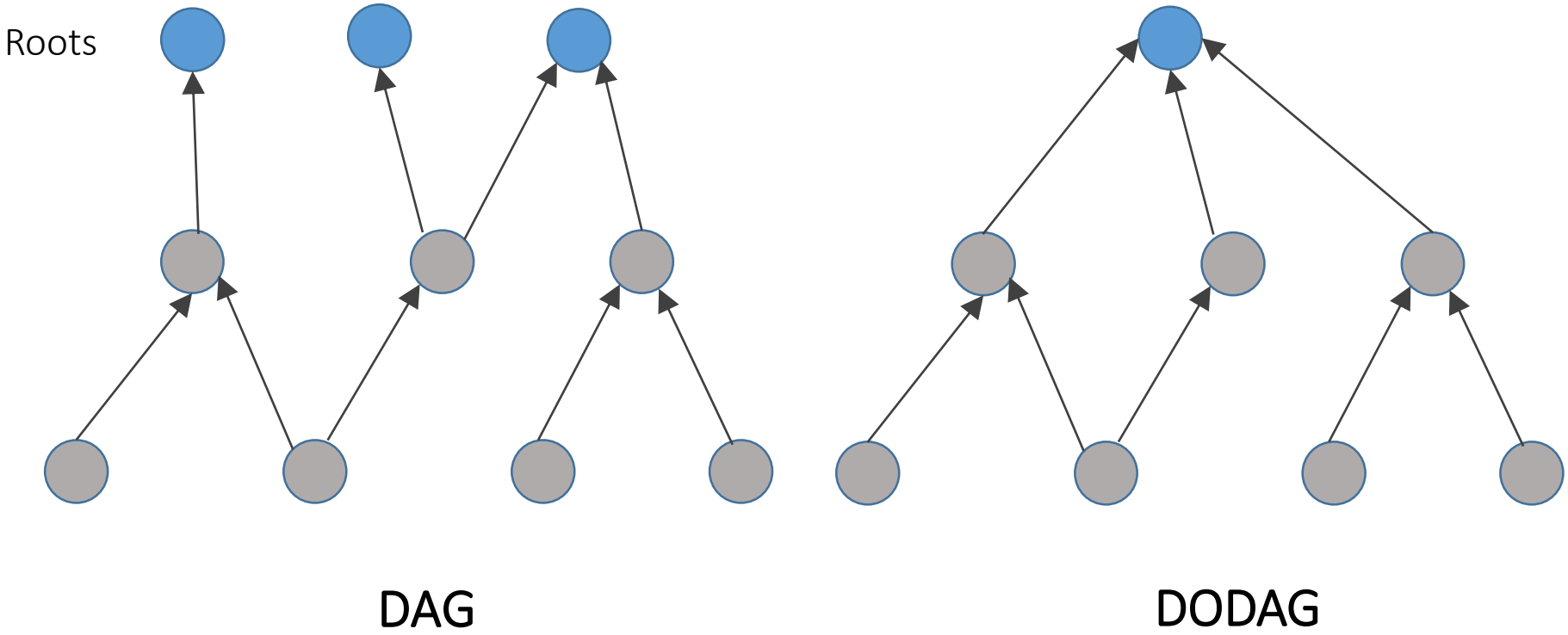
*Texas Instruments: 6LoWPAN demystified



Why not use existing protocols?

- Processing, memory, power constraints
- Single metric not always appropriate for all scenarios (latency vs reliability vs energy)
- Multiple routing instances on the same physical infrastructure make sense for different applications
- Potential point-to-multi-point traffic, many devices
- Directed Acyclic Graph (DAG) topology created to avoid cycles
- RPL actually creates Destination Oriented DAGs (DODAGs), i.e. with a single root

DAG vs DODAG



DODAG construction

- Nodes send link-local multicast DAG information objects (DOI) – configuration + parent discovery
- Parents chosen to minimise the cost of path to the DODAG root
- Nodes listen for DOIs and decide whether to join a new DODAG, or to maintain one already existing
- Sometimes DOI requested via a DAG information Solicitation (DIS)

Routing

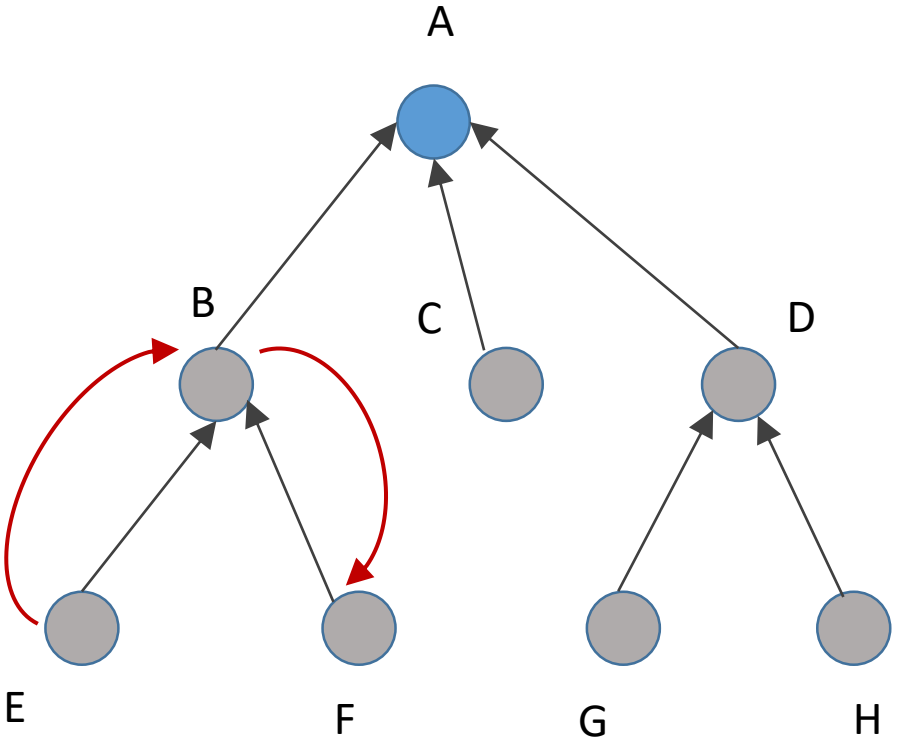
- Each node has a rank relatively to the root (R=0)
- This can be number of hops (distance), expected transmission count (ETX), other
- Upwards routes are towards nodes with a lower rank
- Downwards routes are towards node of increasing ranks

- Many-to-one communication: upwards
- One-to-many communication: downwards
- Point-to-point communication: upwards-downwards

Modes of operation (I)

Storing: each nodes maintain routing table with

- mappings between all destinations reachable via its sub-DODAG and
- Their respective next hop node

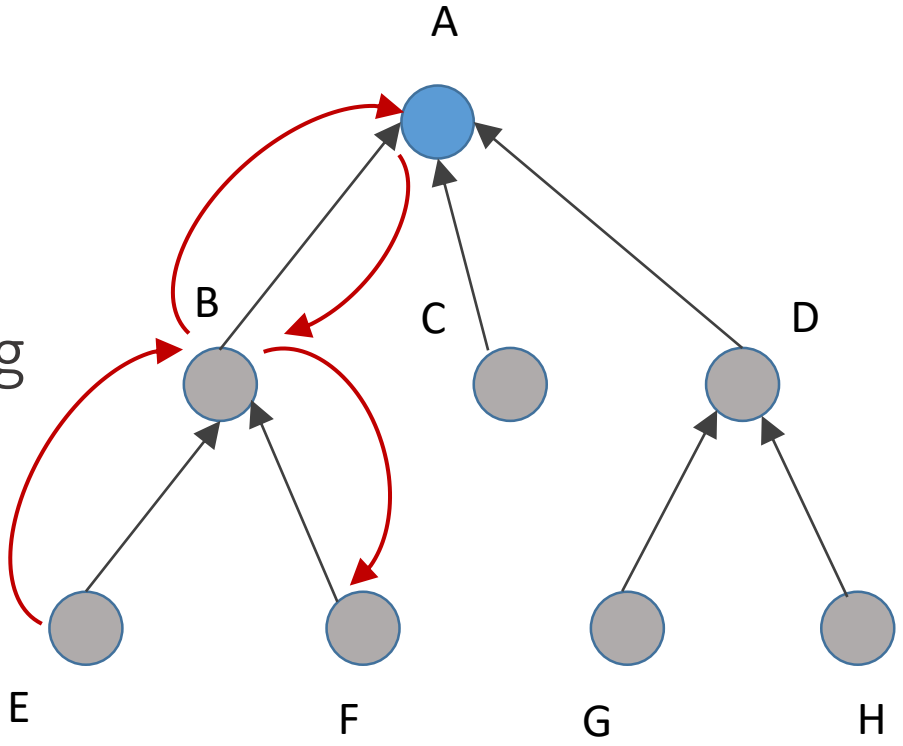


Route: E -> B -> F

Modes of operation (II)

Non-storing:

- Only the root maintains routing information;
- Exploits this by including the information in the packet itself



Route: E -> B -> A -> B -> F