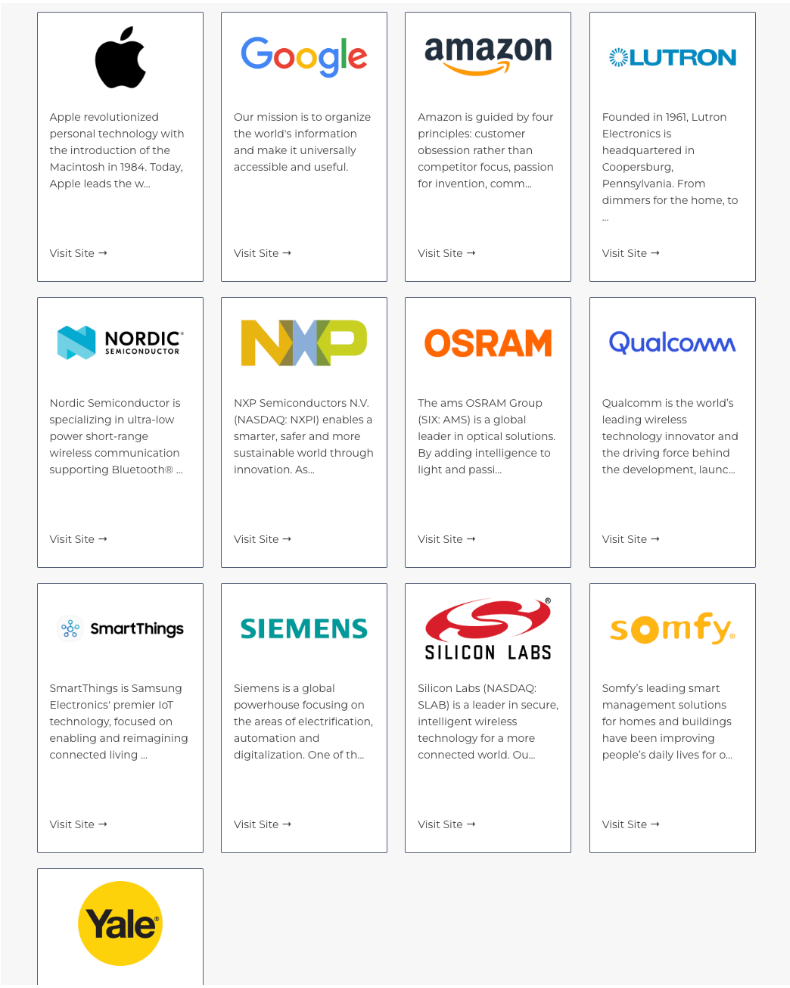
IP-based networking for the IoT

THREAD https://www.threadgroup.org/



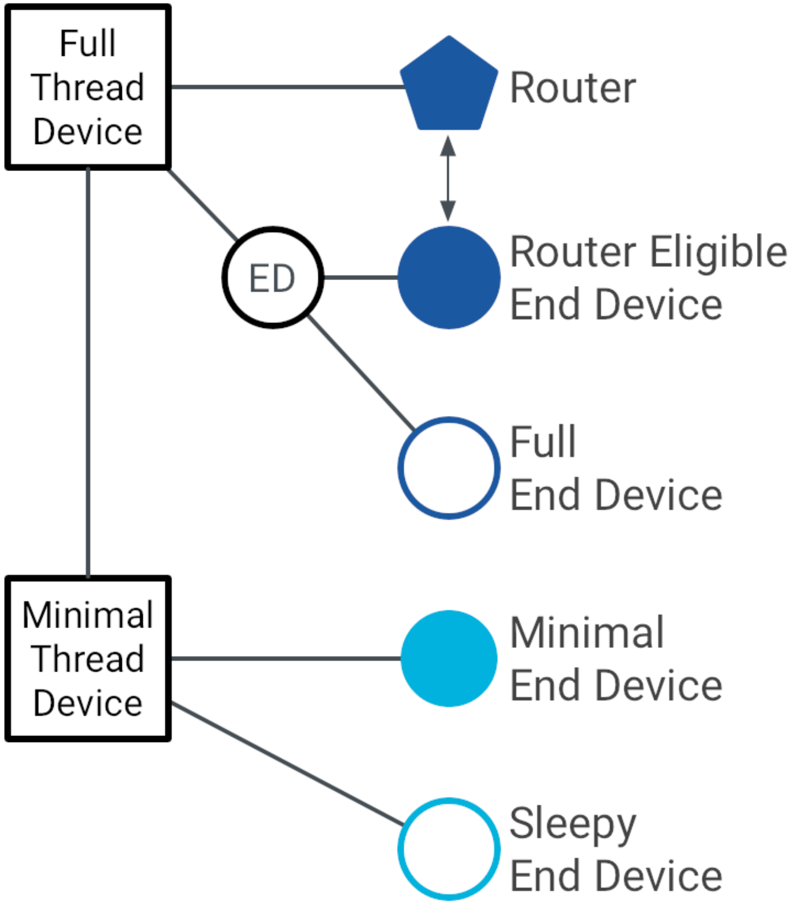
A low-power and low-latency **wireless mesh networking** protocol built using open and proven standards.

Protocol Stack

THREAD	Standard
Application Layer	
UDP + DTLS	RFC 768, RFC 6347, RFC 4279, RFC 4492v RFC 3315, 5007
Distance Vector Routing	RFC 1058, RFC 2080
6LowPAN (IPv6)	RFC 4944, RFC 4862, RFC 6775
IEEE 802.15.4 MAC (including MAC security)	IEEE 802.15.4 (2006)
IEEE 802.15.4 PHY	IEEE 802.15.4 (2006)

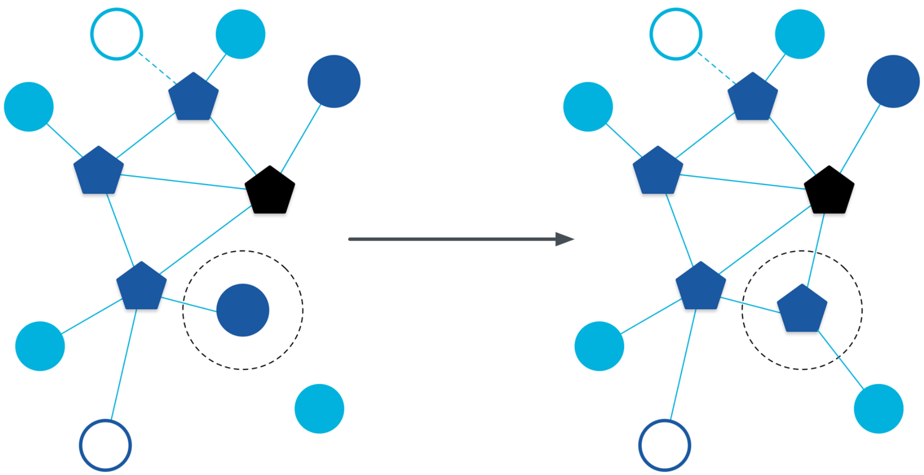
THREAD Devices – types and roles

- Full Thread Device
  - ***Router***: route packets, always active,
  - ***Leader*** elected role of one Router, always active
  - ***Routing-eligible End Device*** (REED): can become router
  - ***Full End Device*** (FED)
- Minimal Thread Device
  - ***Minimal End Device*** (MED), always on
  - ***Sleepy End Device***, duty cycling
  - ***Synchronized Sleepy End Device***, duty cycling at scheduled time
- Boarder Router



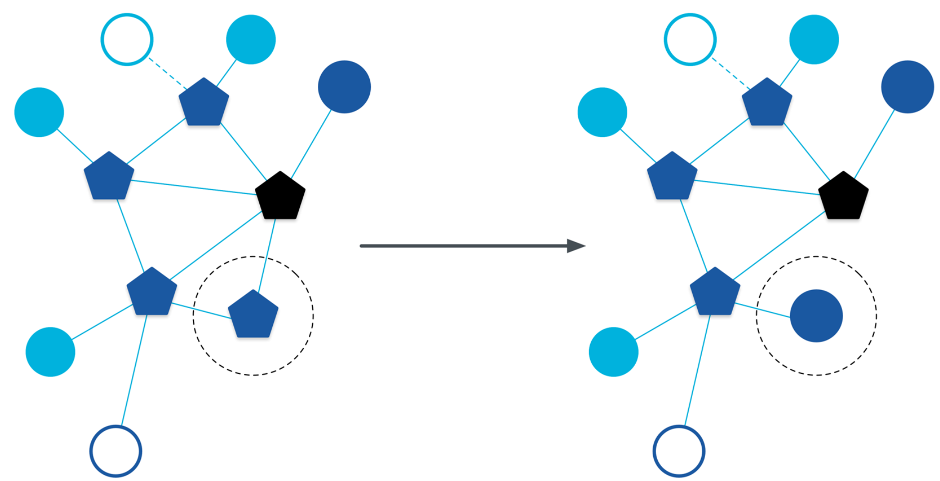
(Synchronized sleepy end device)

THREAD Topology may change

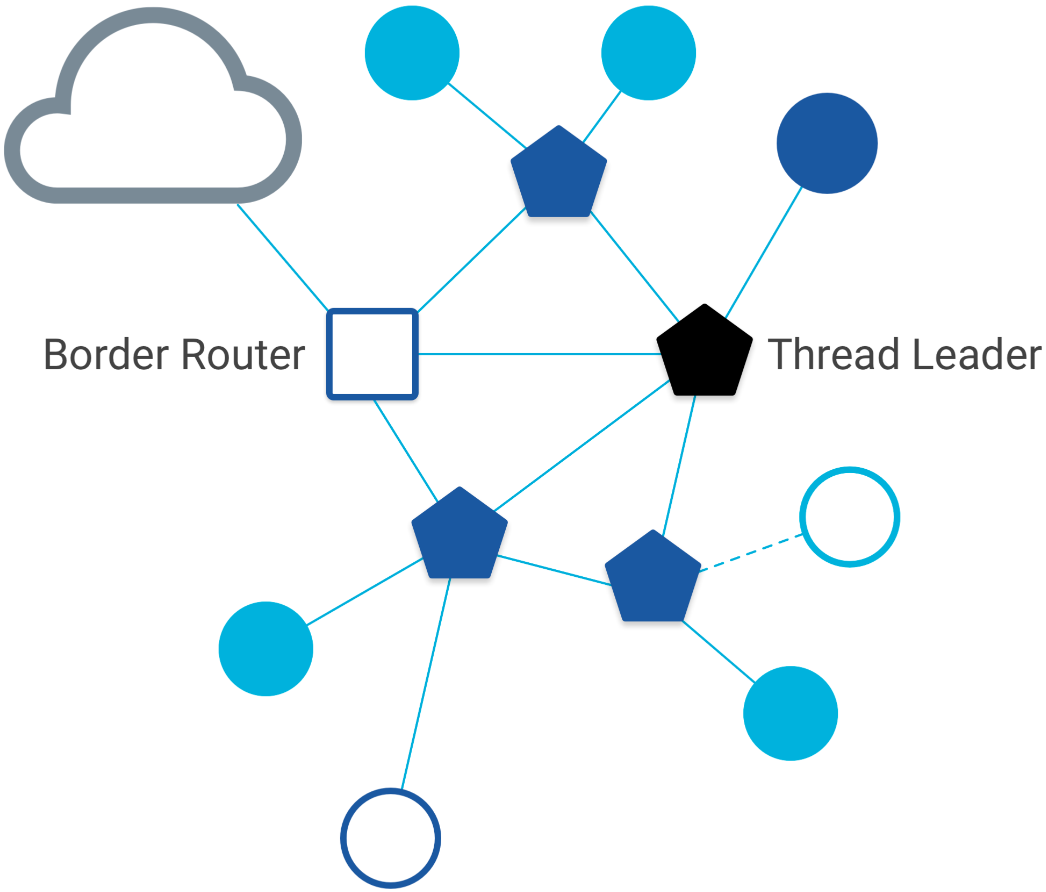


REED can be upgraded to operate as a Router

When a Router has no children, it can downgrade itself and operate as an End Device



THREAD Leader and Border Router



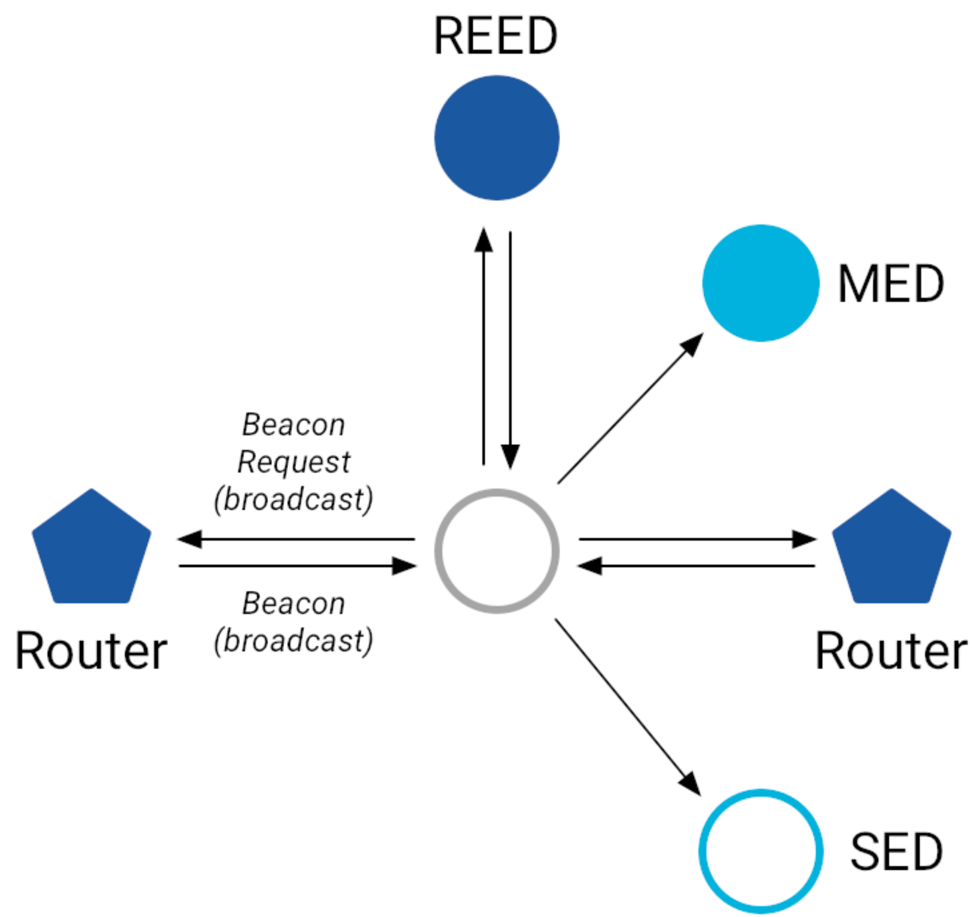
Thread Leader is dynamically self-elected for fault tolerance, aggregates and distributes network-wide configuration information.

A **Border Router** is a device that can forward information between a Thread network and a non-Thread network (for example, Wi-Fi).

- THREAD network identified by:
- 2-byte Personal Area Network ID (PAN ID)
- 8-byte Extended Personal Area Network ID (XPAN ID)
- A human-readable Network Name

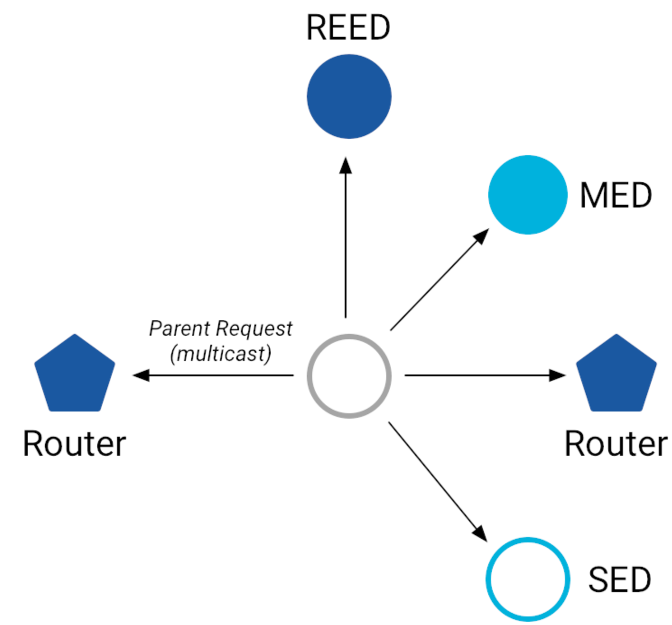
THREAD Network Discovery

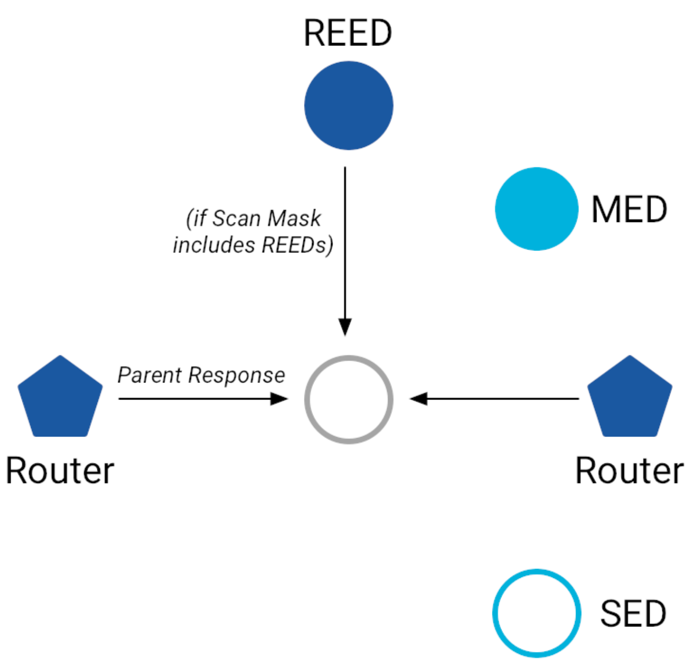
1. The device broadcasts an 802.15.4 Beacon Request on a specific Channel.
2. Any Routers (or REED) in range broadcast a Beacon that contains their Thread network PAN ID, XPAN ID, and Network Name.
3. The device repeats the previous two steps for each Channel



THREAD: Joining a Network (1)

The Child sends a multicast <u>Parent Request</u> to all neighboring Routers and REEDs in the target network.

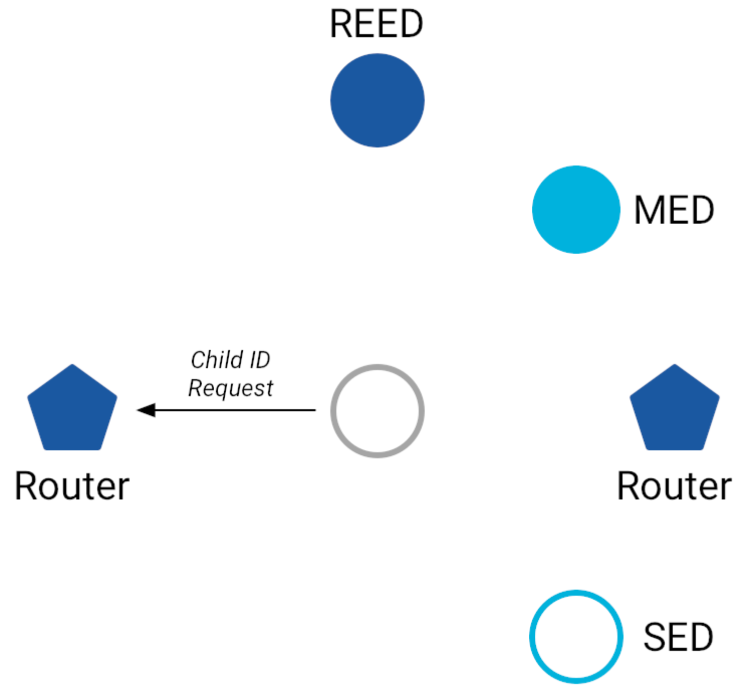


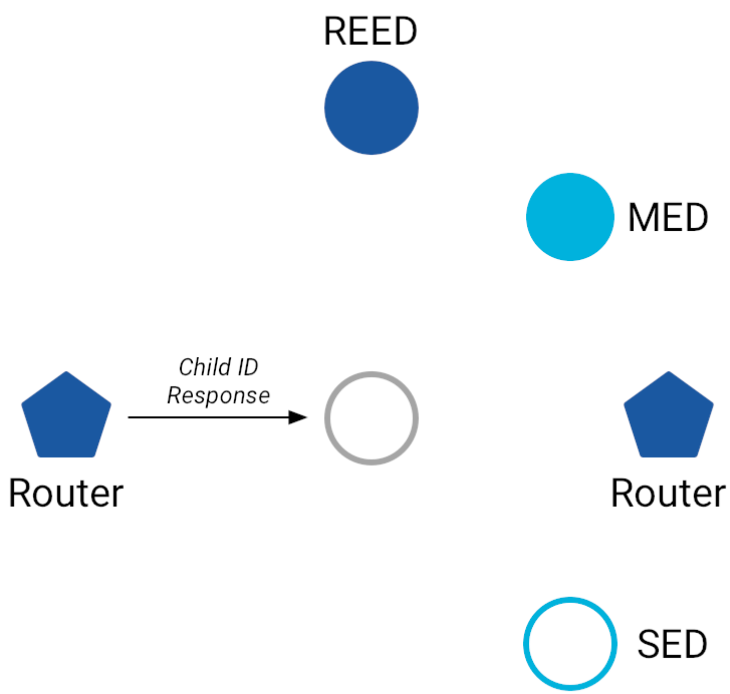


All neighboring Routers and REEDs (if the Parent Request Scan Mask includes REEDs) send Parent Responses with information about themselves.

THREAD: Joining a Network (2)

1. The Child chooses a Parent device and sends a Child ID Request to it.





The Parent sends a Child ID Response to confirm link establishment. (source/destination RLOC, info about THREAD Network)

Internet Protocol v6

- IPv6 (RFC 2460) = the next generation Internet Protocol
  - Complete redesign of IP addressing
  - Hierarchical 128-bit address with decoupled host identifier
  - Stateless auto-configuration
  - Simple routing and address management
- Majority of traffic not yet IPv6 but...
  - Most PC operating systems already have IPv6
  - Governments are starting to require IPv6
  - Most routers already have IPv6 support
  - So the IPv6 transition is coming
    - 1400% annual growth in IPv6 traffic (2009)

IPv4 vs. IPv6 Addressing

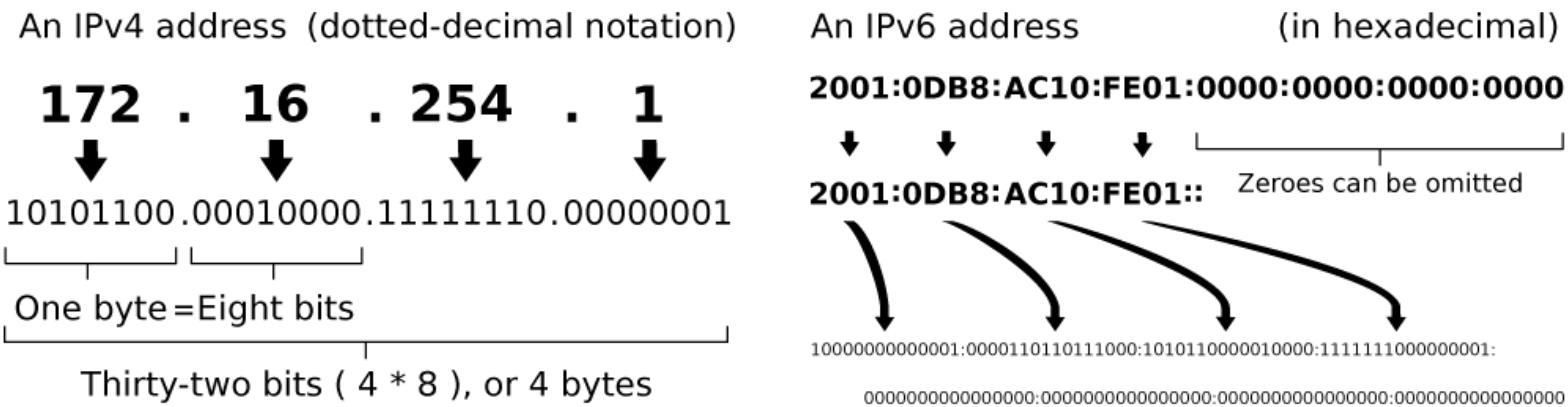
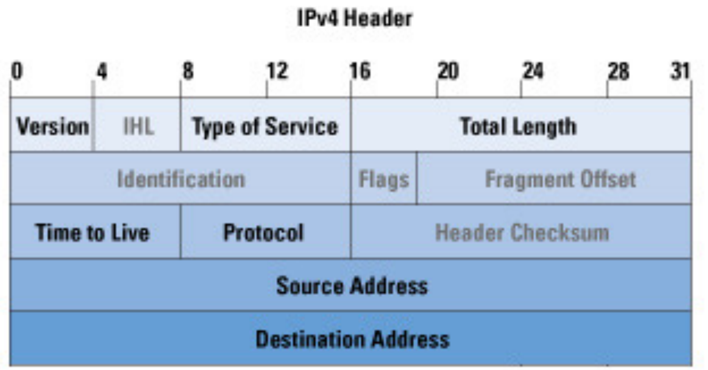


Image source: Indeterminant (Wikipeida)
GFDL

IPv4: 7×10^{-6} [addresses/m^2]

IPv6: 666×10^{21} [addresses/m^2]

IPv4 vs. IPv6 Header



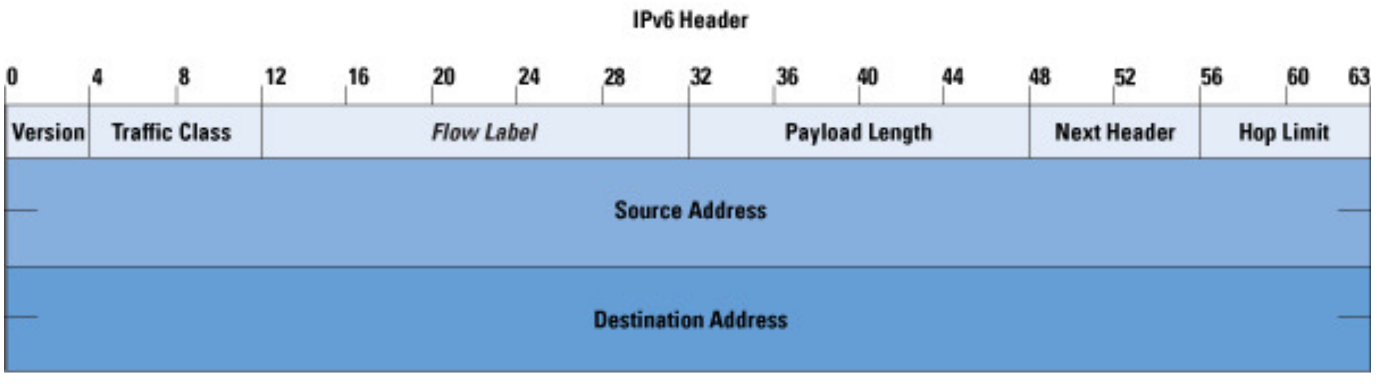


Image source: Bino1000, Mkim (Wikipeida) [GFDL]

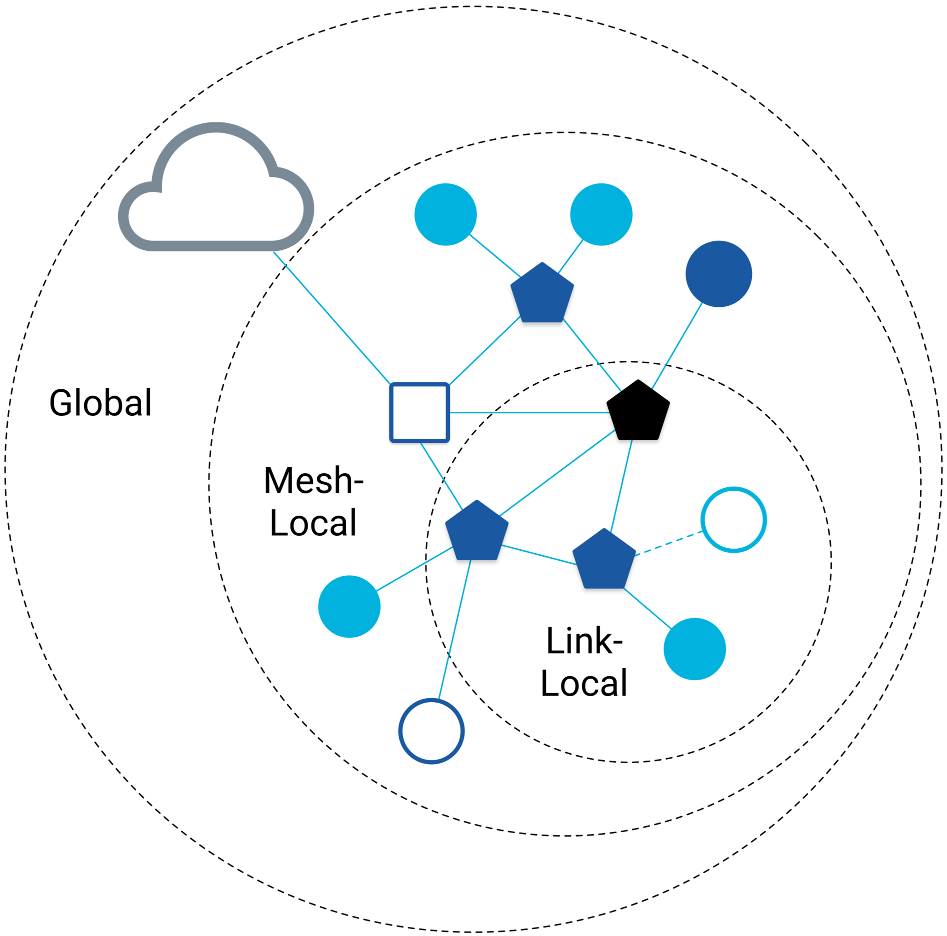
IPv6 Addressing

- 128-bit IPv6 address = 64-bit prefix + 64-bit Interface ID (IID)
- The 64-bit prefix is hierarchical
  - Identifies the network you are on and where it is globally
- The 64-bit IID identifies the network interface
  - Must be unique for that network
  - Typically is formed statelessly from the interface MAC address
    - Called Stateless Address Autoconfiguration (RFC2462)

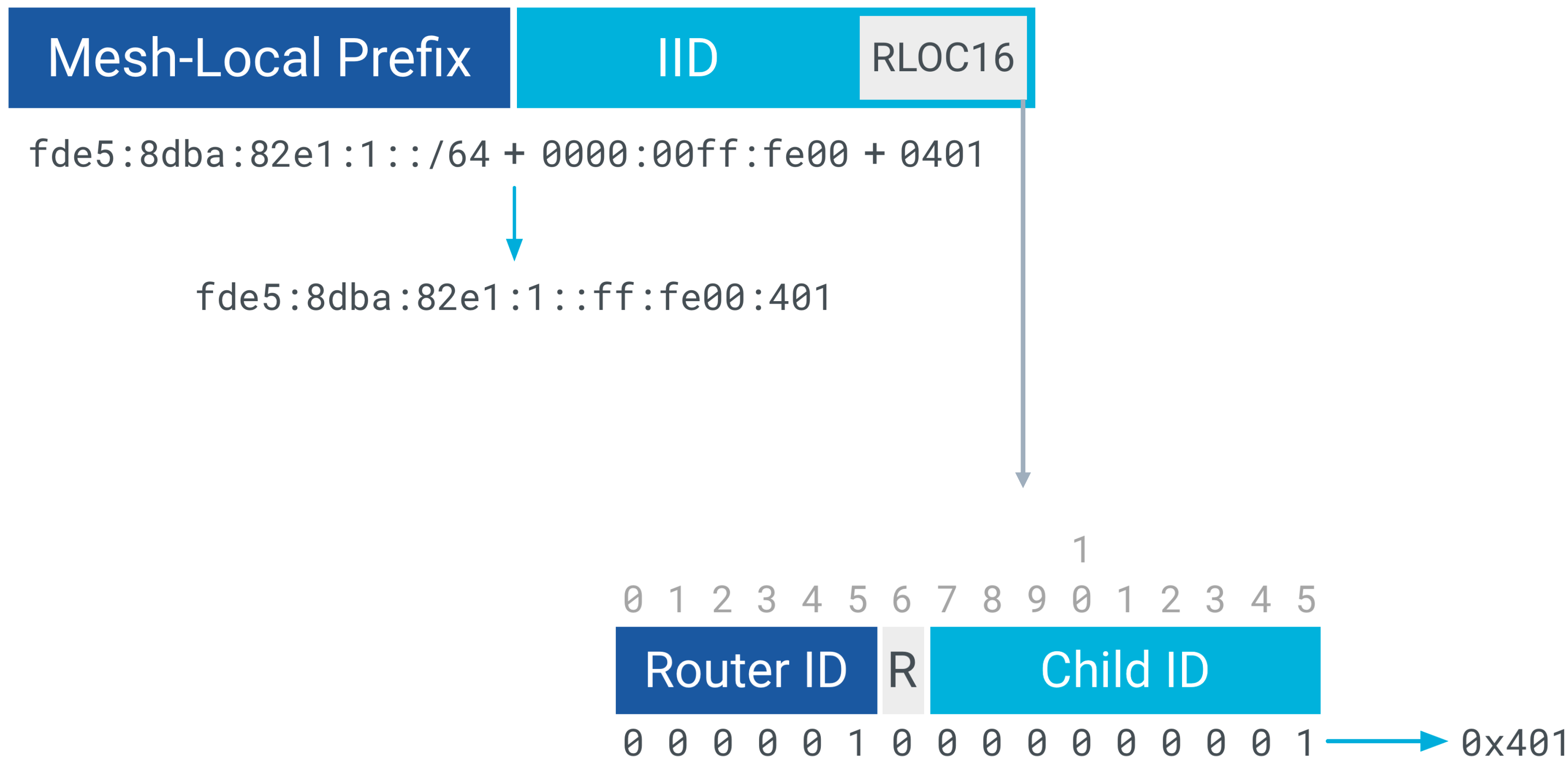
- There are different kinds of IPv6 addresses depending on **scope and context**
  - Loopback (0::1) and Unspecified (0::0)
  - Unicast with global (e.g. 2001::) or link-local (FE80::) scope
  - Multicast addresses (starts with FF::)
  - Anycast addresses (special-purpose unicast address)

IPv6 Addresses in THREAD

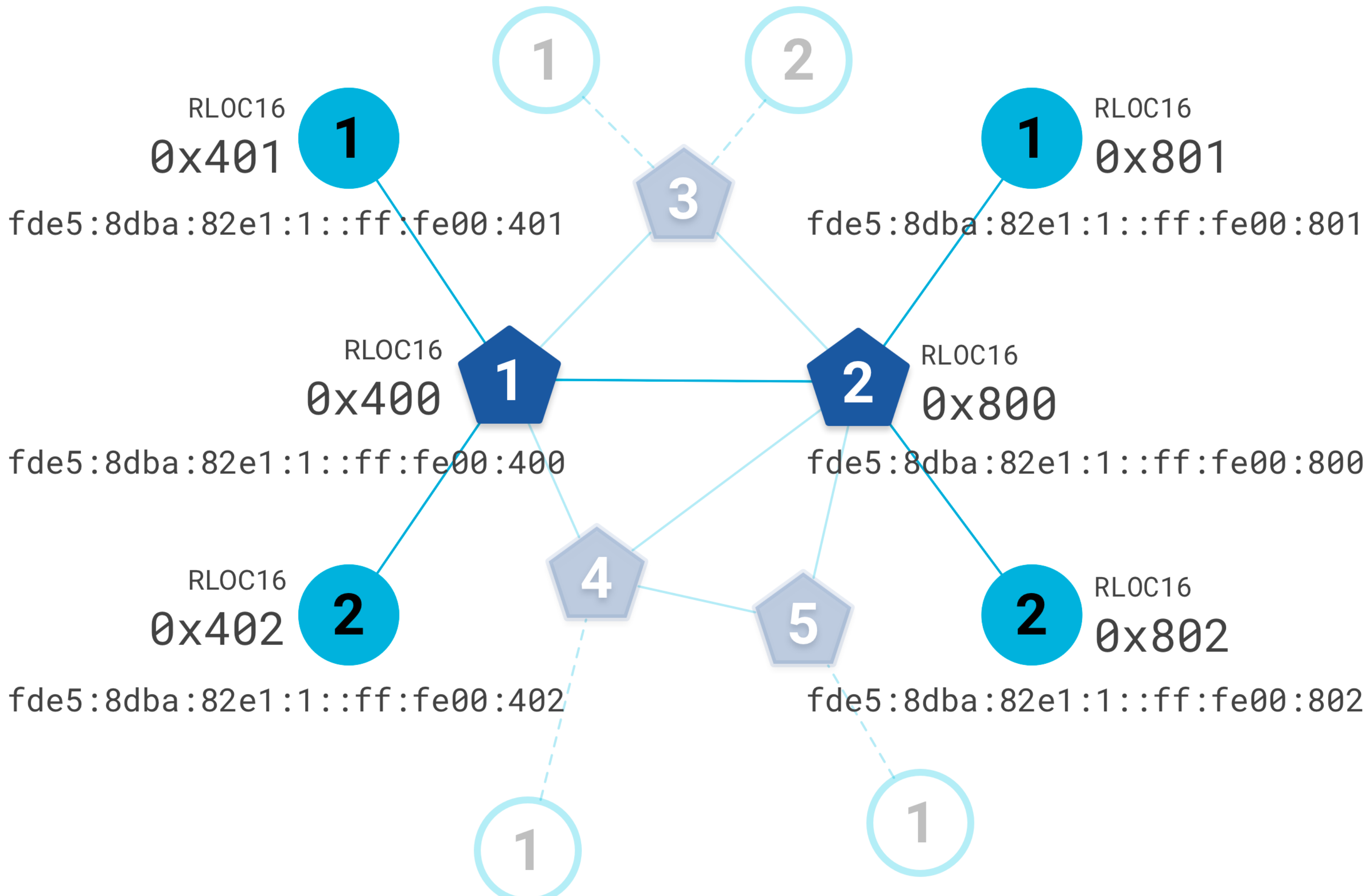
- Scope
  - Link local: interfaces reachable by direct transmission fe80 : :/16
  - Mesh local: interfaces reachable within the same Thread network fd00 : :/8
  - global



IP Addresses in THREAD – Routing Locator (RLOC)



Addressing Example - RLOC



Other THREAD Addresses

- Link Local Address (LLA)
  - Used to discover neighbors, configure links, and exchange routing information
  - Not a routable address
  - Based on IEEE802.15.4

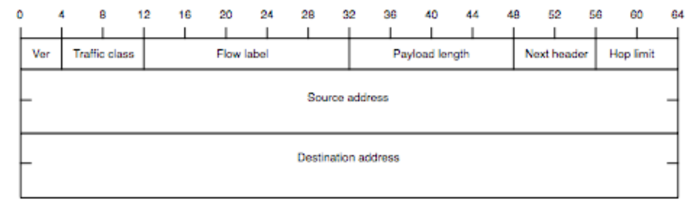
- Mesh Local EID (ML-EID)
  - Random, chosen after commissioning is complete
  - Does not change as the topology changes
  - Should be used by applications

- Anycast Locator (ALOC)

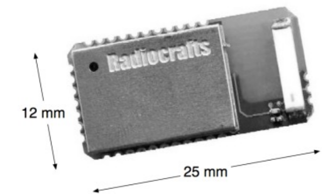
- Global Unicast Address (GUA)
  - A public IPv6 address
  - IID manually assigned or DHCP
  - Always has a prefix of 2000::/3

What is 6LoWPAN?

- An Adaptation Layer to fit IPv6 over Low-Power wireless Area Networks
- Defined by IETF standards
  - RFC 4919, 4944
  - draft-ietf-6lowpan-hc and -nd
  - draft-ietf-roll-rpl
- Efficient header compression
  - IPv6 base and extension headers, UDP header
- Fragmentation
  - 1280 byte IPv6 MTU -> 127 byte 802.15.4 frames

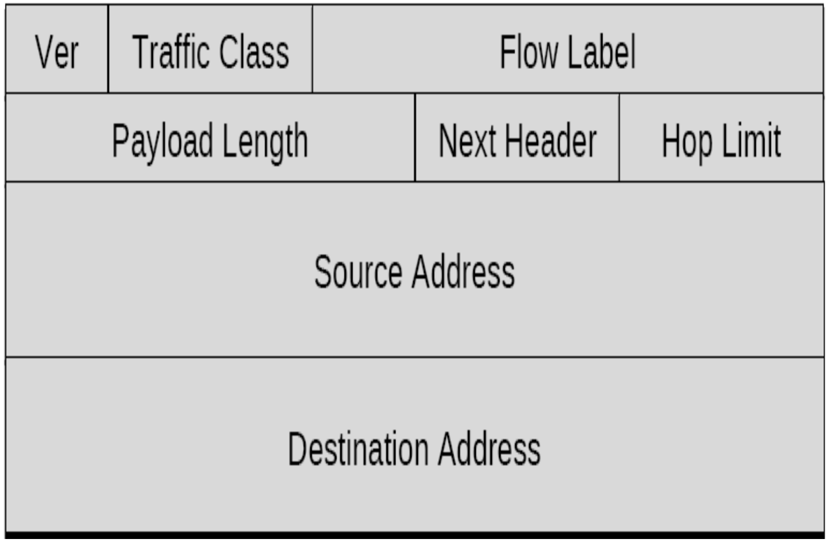


IPv6



6LowPAN Header Compression at a Glance

- Stateless compression
- Flow-independent compression
- Simple tricks on IPv6/UDP header
  - Common values for header fields => compact forms
  - Version is always 6
  - Traffic Class and Flow Label are zero
  - Payload Length always derived from L2 header
  - Source and Destination Addresses can be elided (link-local) and/or compressed depending on the “context” of the transmission

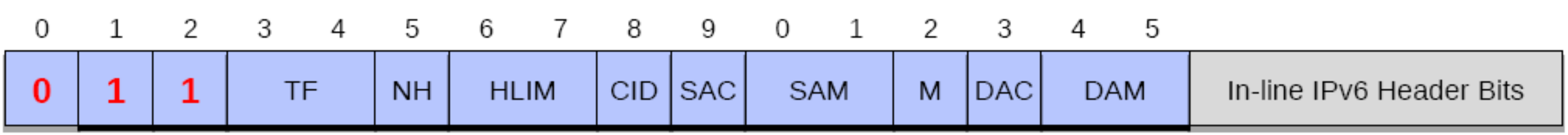


6LoWPAN Addressing

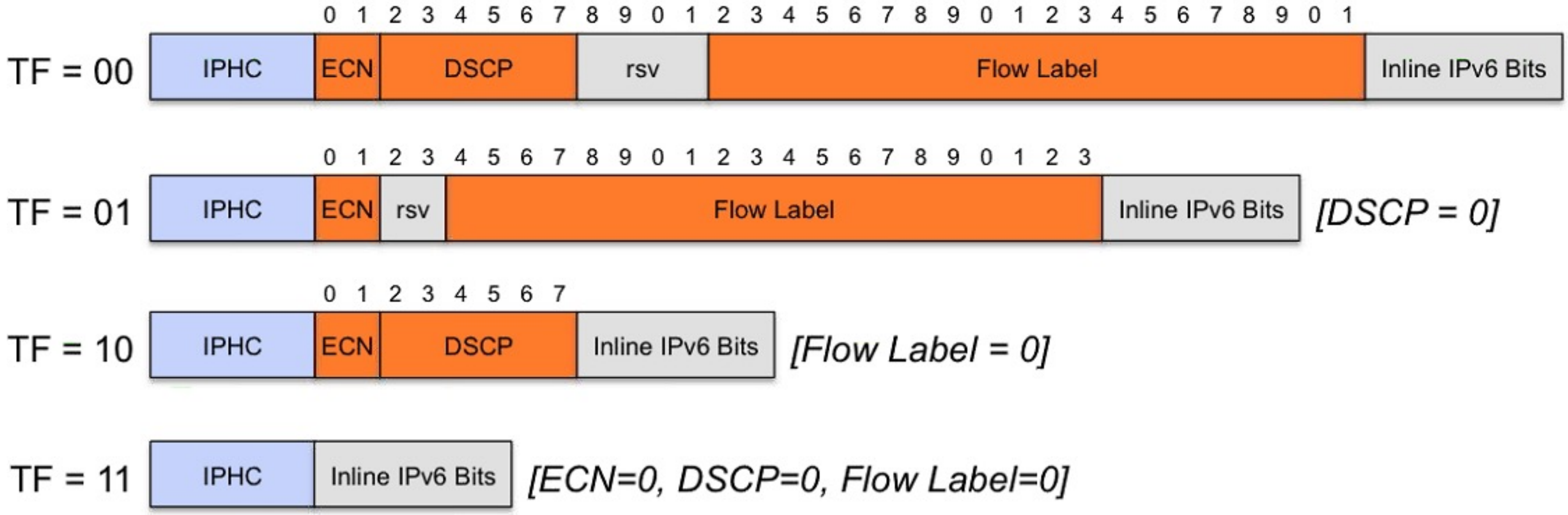
- IPv6 addresses are compressed in 6LoWPAN
- Prefix
  - Addresses within 6LoWPAN typically contain common prefix
  - Nodes typically communicate with one or few central devices
  - Establish state (contexts) for such prefixes – only state maintenance
  - Support for up to 16 contexts 6LoWPAN compresses IPv6 addresses by
- Interface ID
  - Typically derived from L2 addr during autoconfiguration
  - Elide when Interface Identifier can be derived from L2 header

The Header Compression header

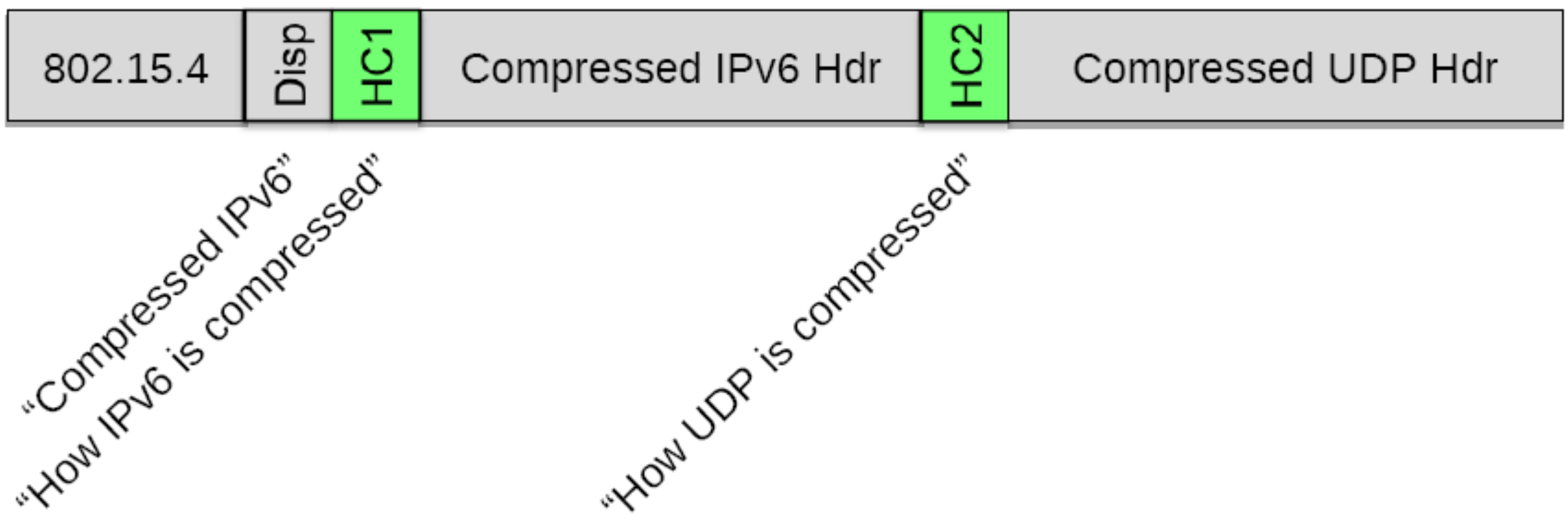
- TF (Traffic Class and Flow Label)
  - 0: Carried Inline (ECN+DSCP+Flow), 1: ECN+Flow, 2: ECN+DSCP, 3: All zero
- NH (Next Header compression)
  - 0: Carried Inline, 1: Next Header is compressed
- HLIM (Hop Limit = Inline, 1, 64, 255)
  - 0: Carried Inline, 1: 1, 2: 64, 3: 255
- CID (Context Identifier Extension)
  - 0: No 1-byte CID identifier, 1: 1-byte identifier follows
- SAC/DAC (Source/Destination Address Compression)
  - 0: Stateless, 1: Context-based
- SAM/DAM (Source/Destination Address Mode)
  - 0: 16 bytes inline, 1: 8 bytes inline, 2: 2 bytes inline, 3: elided
- M (Multicast Destination)
  - 0: Destination is not multicast, 1: Destination is multicast



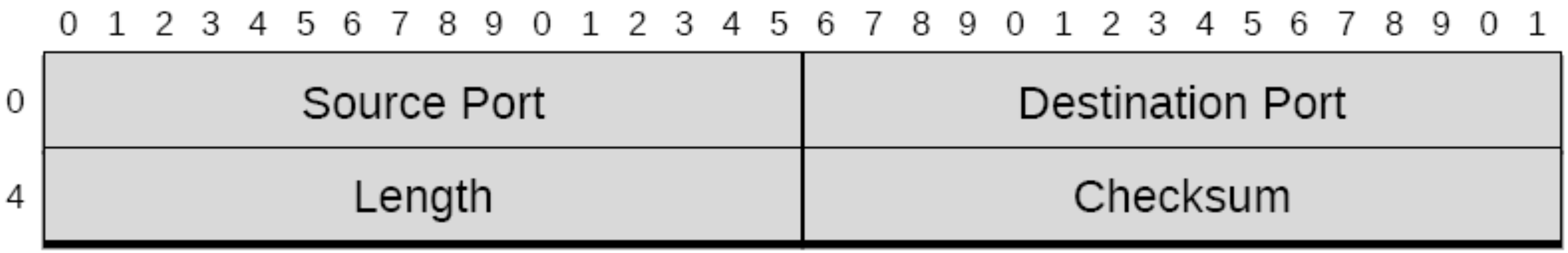
Traffic Class and Flow Label Compression



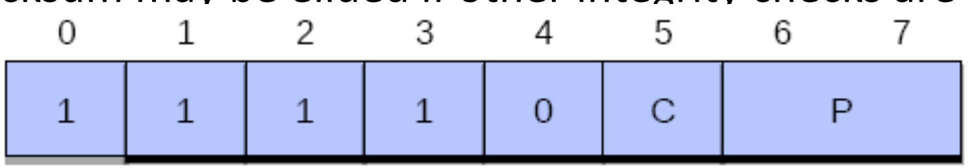
UDP Header Compression



UDP Header Compression

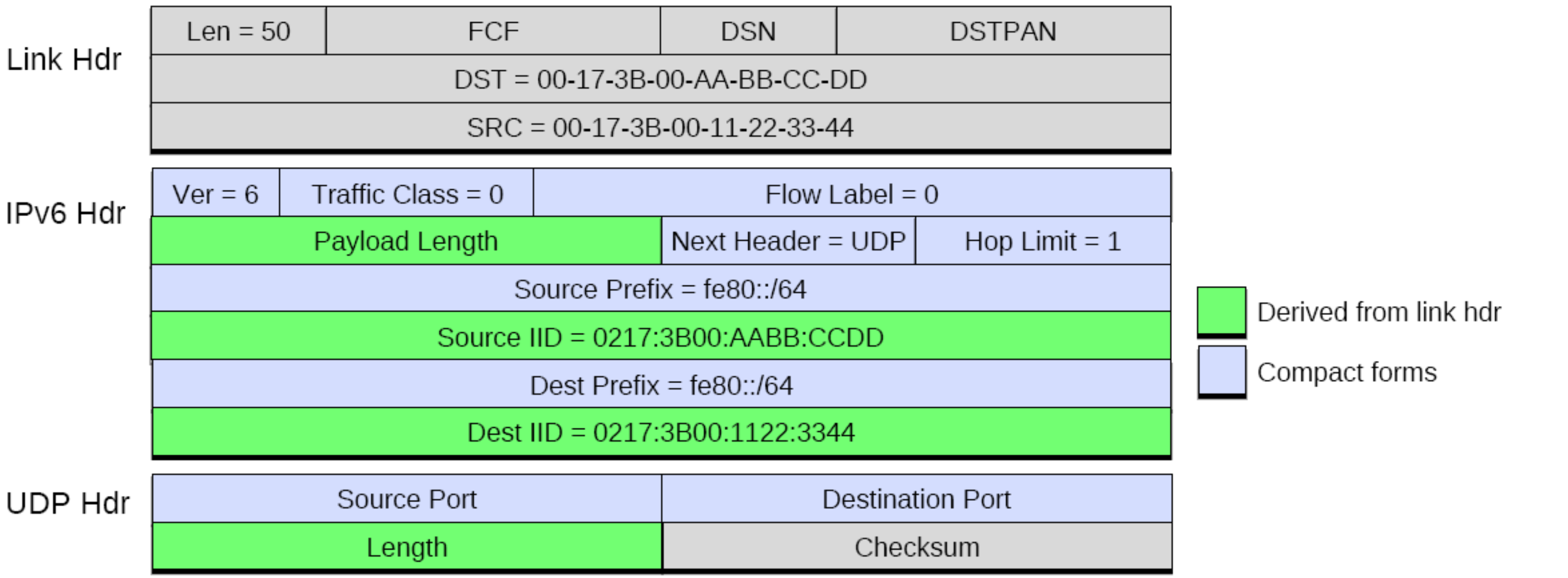


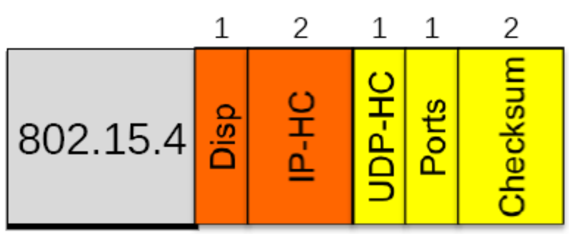
- Assume common values for header fields and define compact forms
  - Ports within 61616 to 61632 (4 bits)
  - Length derived from IPv6 Length
  - Checksum may be elided if other integrity checks are in use (e.g. Ipsec)



  - C (Checksum): 0: Inline, 1: Elide
  - P (Ports):
    - 0: Inline
    - 1: Elide first 8 bits of Dest Port
    - 2: Elide first 8 bits of Source Port
    - 3: Elide first 12 bits of Source and Dest Ports

Link Local

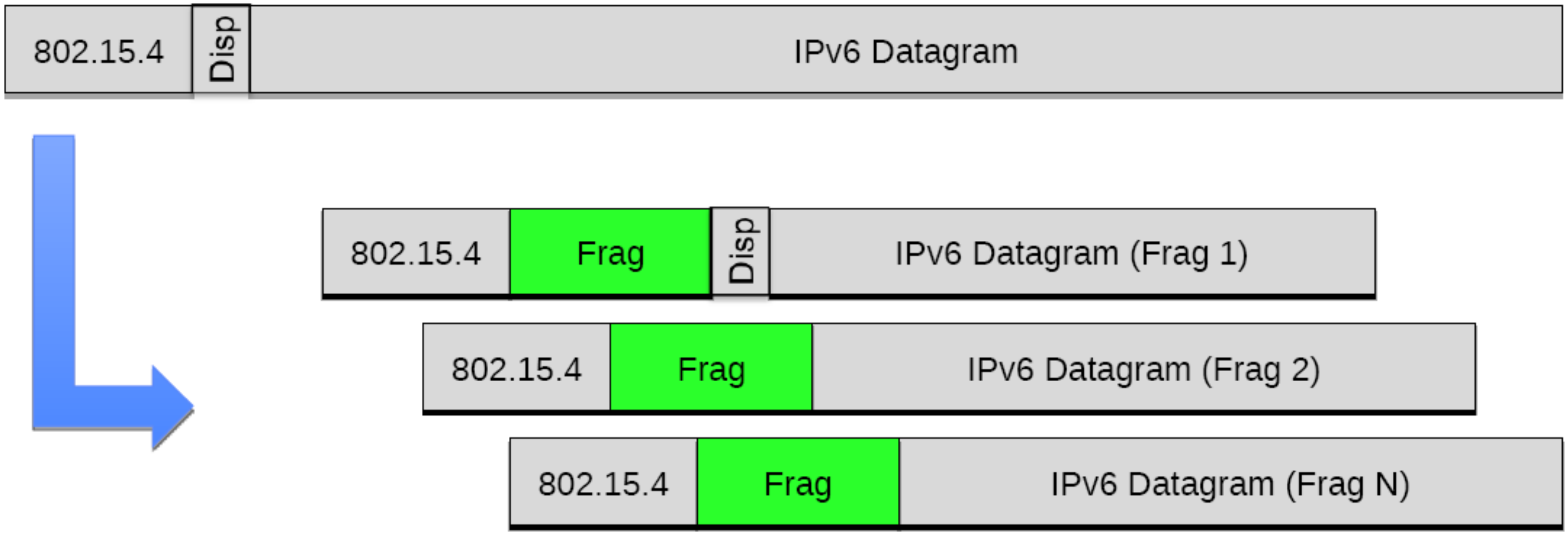




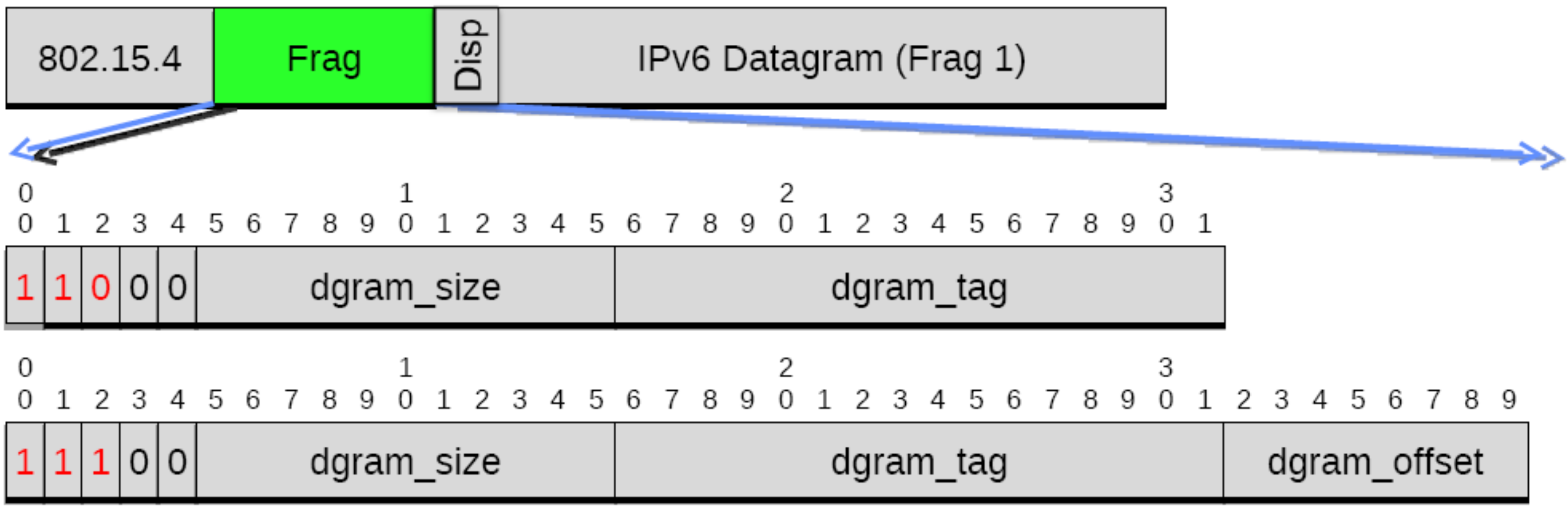
48-byte UDP/IPv6 Hdr→ 7 bytes

Fragmentation

- IPv6 requires a minimum 2-PDU of 1280 bytes
- 802.15.4 has a maximum payload of 127 bytes
- IPv6 packets should be fragmented by 6LowPAN adaptation layer



Fragmentation Header



- *dgram_size*: size of the fragment in bytes
- *dgram_tag*: fragmentation ID (common to all fragments)
- *dgram_offset*: fragmentation offset (word of 8 bytes). Elided in the 1° fragment

Fragmentation in Practice

- The performance of large IPv6 packets fragmented over low-power wireless mesh networks is poor!
  - Lost fragments cause whole packet to be retransmitted
  - Low-bandwidth and delay of the wireless channel
  - 6LoWPAN application protocols should avoid fragmentation
  - Fragmentation handled at the application layer (COAP)
  - Compression should be used on existing IP application protocols when used over 6LoWPAN if possible

Types of Routing Protocols

- Algorithm classes
  - Distance-vector

    Links are associated with cost, used to find the shortest route. Each router along the path store local next-hop information about its route table.
  - Link-state

    Each node acquires complete information about the network, typically by flooding. Each node calculated a shortest-path tree calculated to each destination.
- Types of Signaling
  - Proactive

    Routing information acquired before it is needed.
  - Reactive

    Routing information discovered dynamically when needed.
- Route metrics are an important factor

The Most Common Routing Metric

- Node
  - Residual Energy (node)
  - CPU, Storage, WorkLoad, Battery/Mains
- Link
  - Throughput (local/global metric)
  - Latency (local/global metric)
  - Reliability (local/global metric)
    - Expected Transmission Count (ETX)
    - Link Quality Level (LQL)
- Hop Count

The ETX

- "The average number of packet transmissions to successfully transmit a packet"
- Ex:

  - Being the packet error probability from A to B and from B to A *p* and *q*, respectively, the ETX is: 1/[(1-p)(1-q)]
  - *p* and *q* can be estimated through periodic beaconing

THREAD routing, Proactive, distance vector

- Each router has a routing data base

  - ***Route id set***: list of routers ID and SN
  - ***Link set***: list of current/recent neighbors

  <l_router_id, l_link_marging, l_incoming_quality, l_outgoing_quality, l_age>

  - ***Route set***: routing table

  <destination, next_hop, route_cost>

- Periodic signalling messages
  - Router ID, incoming_quality, outgoing_quality, route cost, sequence_number

Routing metrics

- Link margin and link quality

Link Margin	Quality	Link Cost
> 20dB	3	1
> 10dB	2	2
> 2dB	1	4
≤ 2dB	0	infinite

- Link margin is obtained through the exchange of periodic messages (64Route TLV advertisment)
- Link margin values are low-pass filtered
- Link quality is min(incoming_quality, outgoing:quality)

Distributed Bellman-Ford

Upon receiving a Route64 TLV (Distance Vector)

1. Update link quality
2. For each destination in *Route64 TLV*
   - If destination not in *route set*
     - Add destination/cost in *route set*
   - Altrimenti
     - If next hop route set is equal to *Rout64 TLV* sender
       - Modify route set entry with new info in Route64 TLV
     - Otehrwise
       - If cost *Rout64 TLV* is lower than the one store in route set
         - Modify route set entry with new info in Route64 TLV
3. Return

Modifica delle tabelle di routing

message from C

Net2	4
Net3	8
Net6	4
Net8	3
Net9	5

message from C after increment

Net2	5
Net3	9
Net6	5
Net8	4
Net9	6

Old routing table

Net1	7	A
Net2	2	C
Net6	8	F
Net8	4	E
Net9	4	F

Updating algorithm

New routing table

Net1	7	A
Net2	5	C
Net3	9	C
Net6	5	C
Net8	4	E
Net9	4	F

Rules

Net1: No news, don’t change
Net2: Same next hop, replace
Net3: A new router, add
Net6: Different next hop, new hop count smaller, replace
Net8: Different next hop, new hop count the same, don’t change
Net9: Different next hop, new hop count larger, don’t change