Reactive Discovery of Point-to-Point Routes in Low Power and Lossy Networks

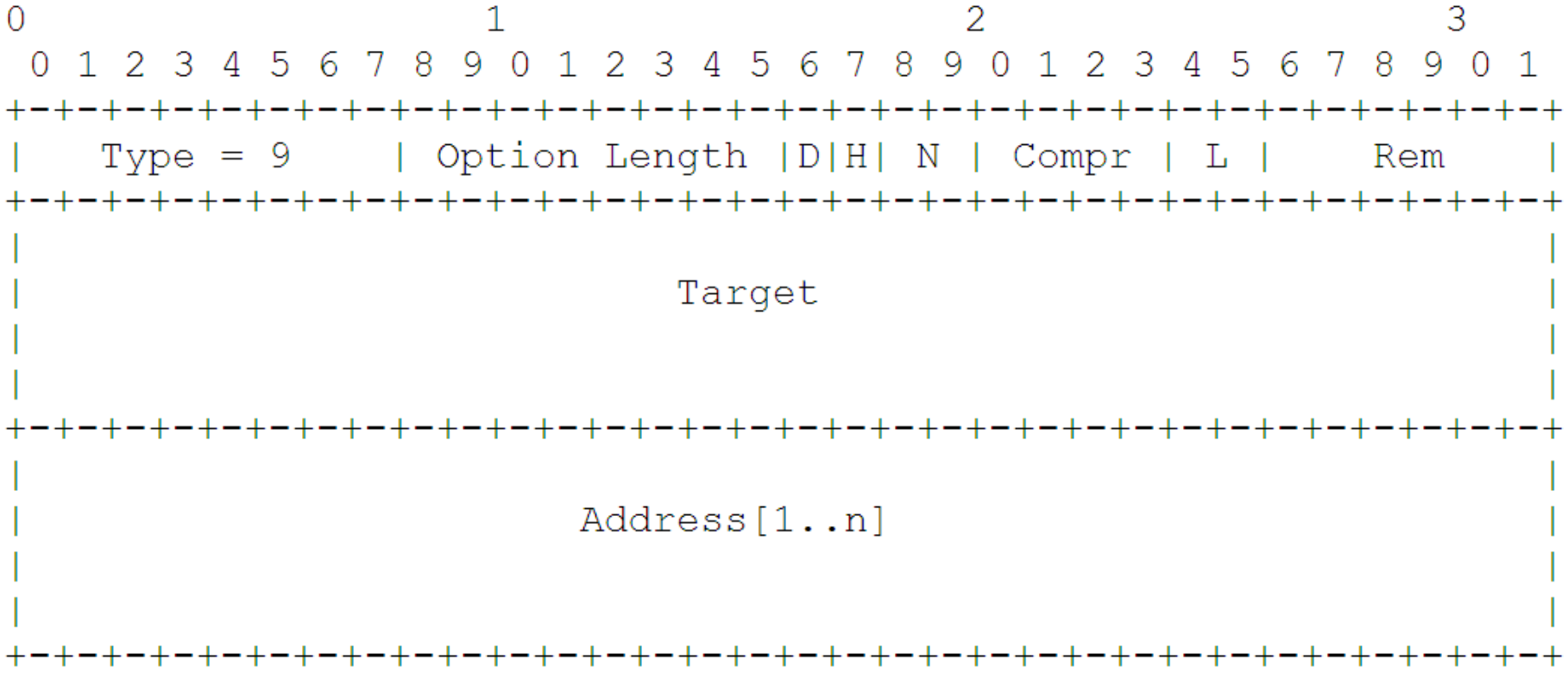
draft-ietf-roll-p2p-rpl-04

Mukul Goyal

University of Wisconsin Milwaukee

Functional Overview

- Origin initiates a temporary DAG.
- The DIOs carries a Route Discovery Option (RDO)



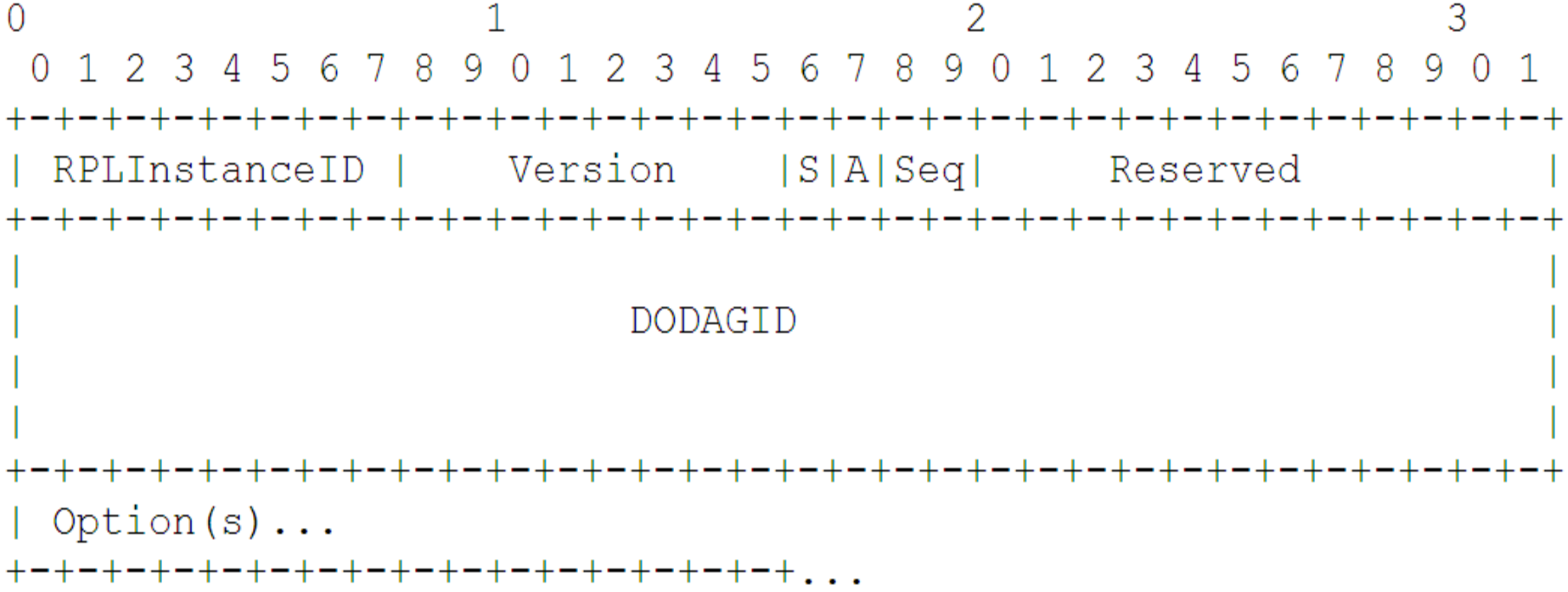
Functional Overview

- The DIOs carry the routing metrics and constraints
  - A DIO is discarded if any mandatory constraint is violated.
- A router lists the best route it knows so far inside the Address vector (inside the RDO)
  - For route diversity, a router SHOULD remember multiple best routes it comes to knows and choose the route to be listed inside DIO randomly.
- The discovered routes are optimized for use in forward direction or in both directions
- The discovered routes MUST have bidirectional reachability
  - To allow Discovery Reply Object (DRO) messages to travel back to the origin
  - Bidirectional reachability is not same as optimizing the route in both directions
- The discovered routes can be hop-by-hop or source routes.
  - Can discover one hop- by-hop route or up to four source routes at a time
  - Can not discover backward hop-by-hop routes
  - The target can use the discovered routes, after reversal, as source routes to reach the origin

Functional Overview

- When the target receives a DIO and if it “selects” the route in there, it does the following
  - If one or more source routes are being discovered, the target sends the discovered route to the origin in a Discovery Reply Object (DRO) message (inside an RDO).
  - If a hop-by-hop route is being discovered, the target sends a DRO to the origin. The DRO travels towards the origin along the discovered route, establishing state for this route in the routers on the path.
  - The target may store the reverse of a discovered route in memory and use it as a source-route to reach the origin.

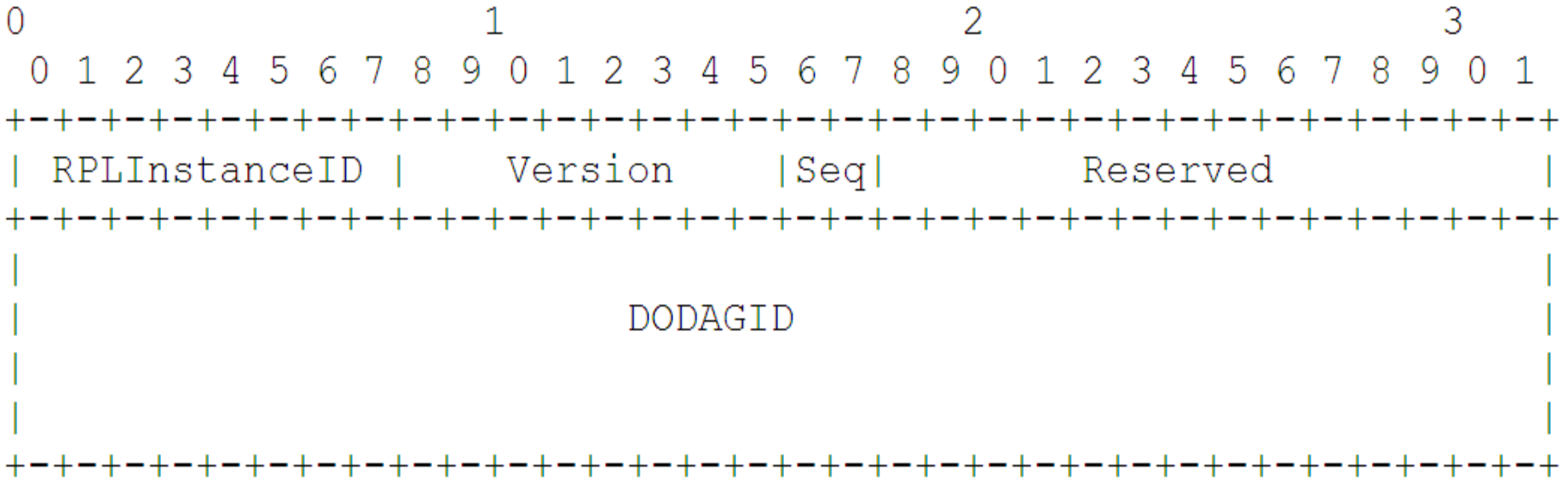
Discovery Reply Object (DRO)



Functional Overview

- The DRO MUST carry an RDO and travel via link-local multicast along the discovered route (listed inside RDO)
  - Only nodes on the route multicast it.
  - If the target has discovered all the routes it needs to, it may set the Stop bit inside the DRO. All nodes receiving such DRO do not transmit any more DIOs for this temporary DAG. They will still process any future DROs they receive.
  - The DROs are susceptible to loss
    - » Cant use link-level acks
    - » No Trickle controlled retransmissions of DROs
    - » Optional end-to-end ack
      - The target may request the origin to acknowledge the receipt of a DRO by sending back a DRO Acknowledgement (DRO-ACK). The origin unicasts a DRO-ACK message to the target. The target resends the DRO if it does not receive the requested DRO-ACK in time.

DRO-Acknowledgement



Trickle Operation For P2P-RPL

- Need low latency for P2P-RPL route discovery
- Then, why use Trickle for P2P-RPL DIO transmissions?
  - Need to avoid redundant DIO transmissions.
  - Need to protect against loss of DIOs due to wireless communication.
- Suggested rules for Trickle for P2P-RPL:
  - The receipt of a DIO, that allows a router to advertise a better route is considered "inconsistent" and hence resets the Trickle timer.
    - The first receipt of a DIO advertising a particular temporary DAG is always considered an inconsistent event under this rule.
  - The receipt of a DIO, that advertises a better route than the router but does not lead to the router advertising a better route itself, is considered "consistent".
  - The receipt of a DIO, that advertises as good a route as the router itself, is considered "consistent".
  - The receipt of a DIO, that advertises a worse route than what the router advertises, is considered neither "consistent" nor "inconsistent", i.e., the receipt of such a DIO has no impact on the Trickle operation.
- The recommended values of redundancy constant "k" is 1.
  - This means that a DIO transmission will be suppressed if the router receives even a single "consistent" DIO during a timer interval.

Packet Forwarding Along a P2P Route

- Use an RH4 header [I-D.ietf-6man-rpl-routing-header] to travel along a P2P source route.
- include an RPL option [I-D.ietf-6man-rpl-option] inside the IPv6 hop- by-hop options header to travel along a P2P hop-by-hop route.
  - The origin MUST set the DODAGID of the P2P route as the source IPv6 address of the packet.
  - The origin MUST specify the RPLInstanceID, associated with the P2P route, inside the RPL option and set the O flag inside the RPL option to 1.
  - A router receiving this packet will check the O flag inside the RPL option and correctly infer the source IPv6 address of the packet as the DODAGID of the hop-by-hop route to be used for forwarding the packet further.

Significant Simplifications Since Last IETF

- What routes can be discovered?
  - one hop-by-hop route or up to 4 source routes from the origin to the target
  - The routes may be optimized in just forward direction or in both directions
  - The routes need to have bidirectional reachability
  - The target may use the reverse of a discovered route as a source route to reach the origin
  - Cant establish backward hop-by-hop routes
- No separate propagation and route constraints.
  - Can use optional constraints if desired.
  - No need to include a Metric Container inside a Route Discovery Option (RDO)
- Routing metrics and OF used for route discovery and temporary DAG formation are now same.
- Specification of rules to operate Trickle for P2P-RPL .
- Route accumulation in RDO
  - Rather than in Reverse Routing Header

Significant Simplifications Since Last IETF

- The temporary DAG is created solely for route discovery and can not be used for routing purposes
- Discovery Reply Object (DRO) travel via link-local multicast along the reverse of a discovered route.
- DRO may have a “stop” bit to cancel DIO transmissions once route discovery is over.
- Introduces optional DRO Acknowledgement.

INRIA's P2P-RPL Implementation

- Based on current development version of Contiki (2.x-20110630).
- Compliant to version 4 of the draft.
- A minimal application with Contiki, 802.15.4, 6LoWPAN, IPv6, and P2P-RPL
  needs 47 KiB memory.
- Experiments have been carried out on a testbed with 15 MSP430-based
  sensor nodes and an average node degree of 3.8
- A video of P2P RPL in action can be found here: http://www.lix.polytechnique.fr/~mphi/p2p/