Routing with RPL

Refresher: Routing Protocols

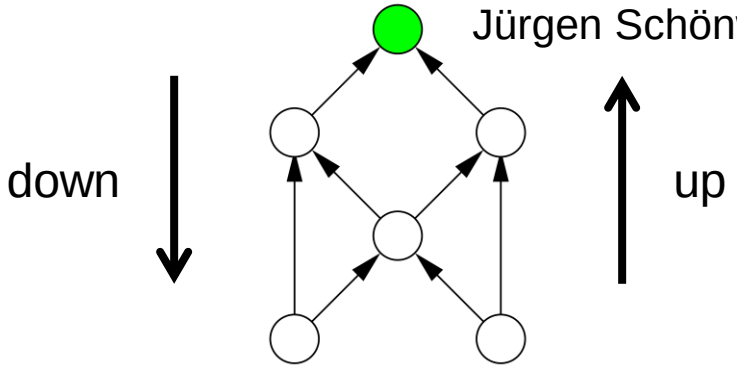
- Two classes of routing protocols:
  - Link State: each node knows complete network and runs a shortest-path algorithm
  - Distance Vector: a node only needs to know which neighbor leads to which node
- Link State Protocols are more powerful but require more resources
- General idea of Distance Vector:
  - Each node maintains for each destination
    - Minimum distance to destination
    - Direction (=neighbor) to forward a packet to destination
  - Initially a node only knows about its neighbors
  - Nodes exchange information with their neighbors

Low-power and Lossy Networks (LLN)

- RPL designed for Low-power and Lossy Networks (RFC 6550)
- Characteristics of such networks:
  - Lossy, slow & unstable links
  - Traffic mostly flows from devices to a server/sink or border router (or in the opposite direction)
  - Large number of nodes
  - Constrained devices with small memory

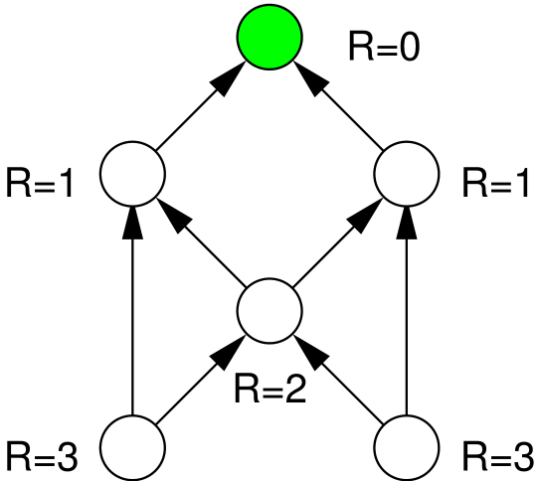
Routing Topology

- Destination Oriented Directed Acyclic Graph (DODAG) where the root is a server, border router, etc.
- Not a tree! Alternative paths possible



- RPL Instance = Multiple disjoint DODAGs
- You can run multiple RPL instances on the same physical network, for example one optimized for energy consumption, one for delay, etc.
- RPL control messages carry a DODAG ID and a RPL-Instance ID

Ranks



Jürgen Schönwälder, Jacobs University Bremen

- Rank is a node property, typically proportional to path cost but coarser (e.g., 100, 200, 300,…)
- Rank of node must be greater than rank of parent

RPL Control Messages: DIO

- DAG Information Object (DIO)
  - Link-local multicasted by existing RPL nodes to advertise a DODAG
  - Sent periodically but also on request (DIS message) or when routing problems detected
  - Contains rank of sending node and network prefix
  - Nodes find parent(s) in this way
  - Multiple parents possible. Choice depends on path costs
- Result:
  - Nodes know about possible parents
  - Routes up ***toward*** DODAG root established

Path cost

- How does a node X choose a parent Y and compute its rank?
- Idea: Based on how “expensive” the path is, e.g.:
  - Signal quality
  - Delay
  - Throughput
  - …
- DODAG can define an *Objective Function* that the nodes should use to choose a parent (RFC 6551+6552)

RPL Control Messages: DAO

- Destination Advertisement Object (DAO)
  - DAO messages travel up
  - Are unicasted to parents (or link-local multicasted to help for direct P2P traffic)
  - Forwarded to root
  - Propagate information about children (new & disappeared) so that parents can populate their routing tables
- Result:
  - Routes down away from root established

RPL Control Messages: DIS

- DODAG Information Solicitation (DIS)
  - Node can send DIS message to request DIO
  - When?
    - A new node wants to join an RPL instance
    - A sleeping node wakes up and needs recent information about the network/wants to inform the network that it is back
    - A node cannot reach the router anymore -> looking for a new parent

Packet Forwarding

- In up direction:
  - Packet is sent to lower rank (or sibling, if no lower rank)
- In down direction:
  - Node knows route to destination thanks to DAO messages
- Note: nodes do not know complete route to destination, only next hop to destination is stored
  - That means: a node knows that to route a packet to destination X, it has to forward it to node Y which is the next hop to X
- Basically, it's a distance vector protocol with ranks (and some other mechanisms discussed later) to avoid loops
- This is called RPL "storing mode"

RPL Lite

- RPL Lite is the default RPL implementation in Contiki-NG
- In RPL Lite, RPL works in “non-storing mode“:
  - Non-storing: routing information is not stored on the nodes
  - Advantage: less memory consumption
- In non-storing mode, only one node in the network (e.g. the root node of the DODAG) stores all routing information
  - When a packet is sent from a node to the root: nothing special happens
  - When a packet is sent from the root to a node: Source routing is used

Source routing (RFC6554)

- When the root wants to send a packet to a node in the DODAG, the root adds information about the route to the packet:

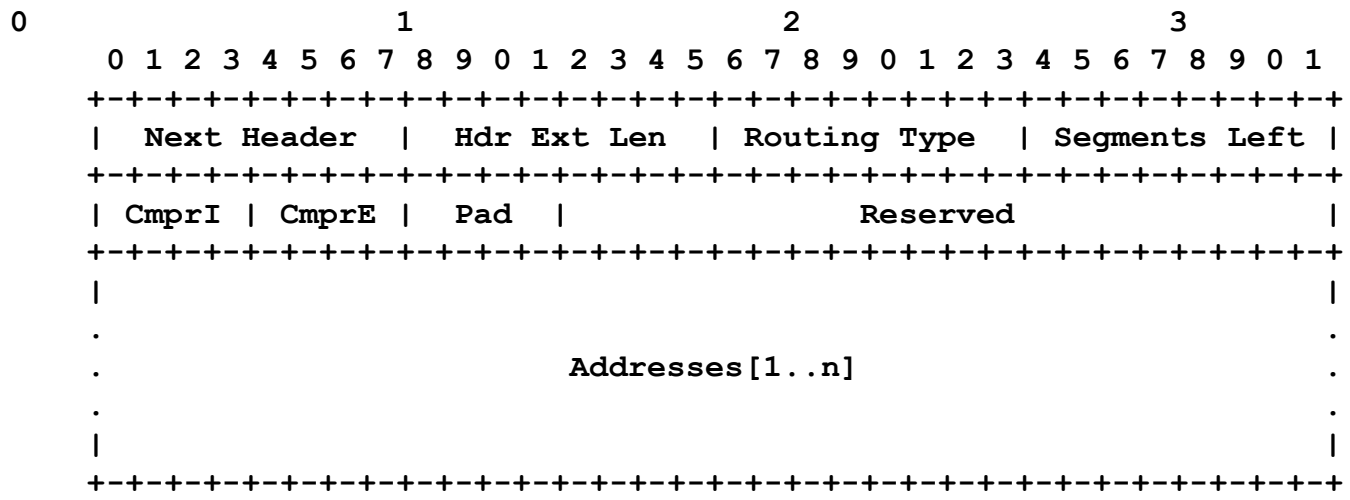
IPv6 header

Source routing header

Packet payload

- The source routing header contains the IP addresses of the nodes for the path from the root to the destination
- When a node receives such a packet, it will check the next hop in the source routing header and forward the packet
- Disadvantage: source routing header can become large if paths are long

Source routing header



- Segments left is initialized to *n*. At each hop, it's decreased by one.
- To save space, the sender can specify the number of bytes of the prefix that the addresses share with the destination address (CmprI and CmprE fields)
  - CmprI = 0: the addresses are all full 64-bit addresses
  - CmprI = x: the entries only specify the lower $16 - x$ bytes of the addresses