RPL

Routing Protocol for Low-power and lossy networks

Introduction

LLN: Low-Power and Lossy Network.

- Typically composed of many embedded devices with **limited power, memory, and processing resources** interconnected by a variety of links, such as IEEE 802.15.4 or low-power Wi-Fi.
- There is a wide scope of application areas for LLNs, including industrial monitoring, building automation (heating, ventilation, and air conditioning (HVAC), lighting, access control, fire), connected home, health care, environmental monitoring, urban sensor networks, energy management, assets tracking and so on.

- Low Power and Lossy Networks (LLN) are resource constrained
- Routers are usually limited in terms of processing power, battery and memory, and their interconnects are characterised by unstable links with high loss rates, low data rates and low packet delivery rates
- The traffic patterns could be P2P or P2MP or MP2P
- Lossy means the packet drop rate will be high.

RPL-Introduction

- RPL is a distance vector routing protocol
- RPL mainly targets collection-based networks, where nodes periodically send measurements to a collection point.
- The protocol was designed to be highly adaptive to network conditions and to provide alternate routes, whenever default routes are inaccessible.
- RPL provides a mechanism to disseminate information over the dynamically formed network topology
- Contains thousands of nodes…

Terminology related to RPL

- **Directed Acyclic graph** : It is a graph that contains no cycle(Figure 1), we see such kind of graphs in Spanning trees.
- **Root** : It is the destination of the node in DAG, it has no outgoing edge.
- **Up**: It is any edge that is directed towards the Root (Figure 2).
- **Down** : It is any edge which is directed away from root.

Figure 1

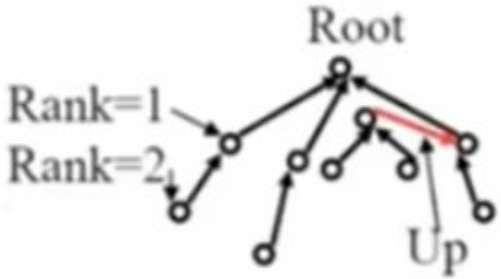


Figure 2

Terminology related to RPL

- **Destination Oriented DAG (DODAG) :** This is a special king of DAG where each node wants to reach a single destination (Figure 1).
- **Objective Function** : It help us to decide whether we are near to the root or away from it. Objective Function is decided by a programmer or designer. It is something which we want to minimize. It can be energy, it can be Latency and once we decide what we want to minimize, we give it a Number.
- **Rank**: As shown in (Figure 1), it is the distance from the Root.
- **RPL Instance** : When we have one or more DODAGs, then each DODAG is an instance. Figure 2 shows two RPL Instances.

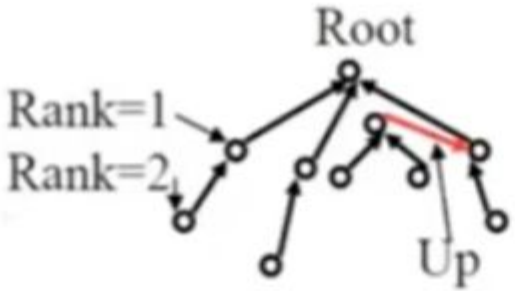


Figure 1

Figure 2

Terminology related to RPL

- **DODAG ID:** Each DODAG has IPv6 (128 bit) this ID is given to its root only and as long as the root doesn't change, ID also doesn't change.
- **DODAG Version:** Each new shape of DODAG means a new version
- **Goal:** It is where a DODAG want to reach. it can be wired network. Goal is different than objective function and objective function our aim is to minimised. However goal is where we want to go
- **Grounded:** When a DODAG reaches its goal it is known as Grounded. G in figure one shows grounded DODAG.
- **Floating**: When a DODAG is not connected or is yet to reach the goal it is called Floating. F in Figure 1 shows Floating DODAG.

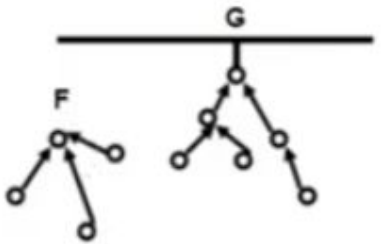


Figure 1

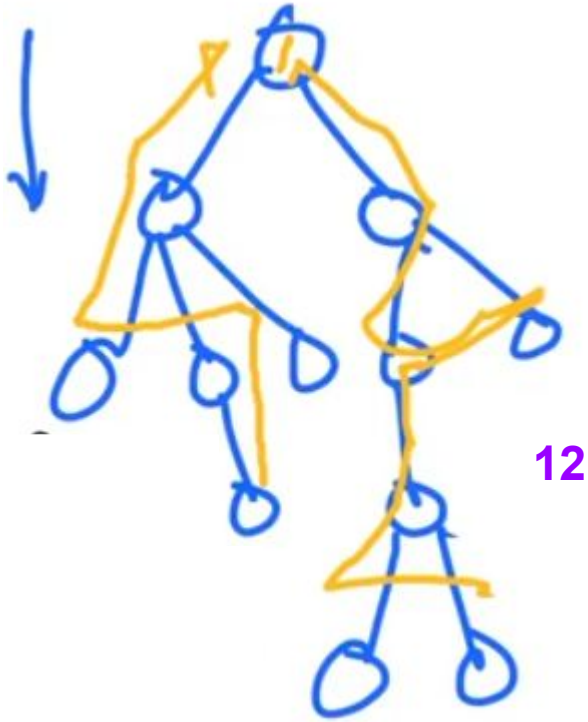
Terminology related to RPL

- **Parent:** Parent is where arrow is pointing towards, and a child is where the arrow comes from. Parents can have multiple children. Similarly, a child can have multiple parents.
- **Sub-DODAG:** It is any subtree of given DODAG.
- **Storing:** Storing nodes keep the whole routing table they know how to go from One node to another.
- **Non-storing:** They are simple they don't restore an entire routing table they only know about their parents.
- The whole DODAG except from the root has to maintain a uniformity, it has to be either storing or non-storing. Root is always storing.

RPL Topology

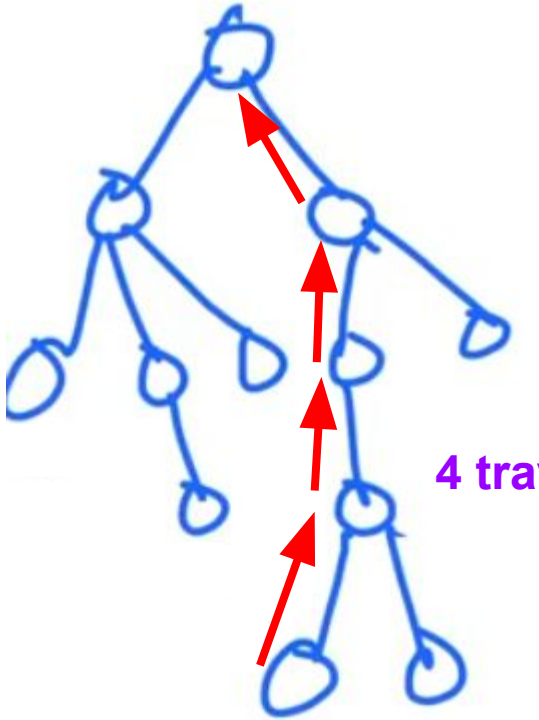
- Upward path is so common (mp2p)
- Downward path is optional mainly for p2p and p2mp
- An RPL Instance consists of multiple Destination Oriented Directed Acyclic Graphs (DODAGs). Traffic moves either up towards the DODAG root or down towards the DODAG leafs

Downward Path



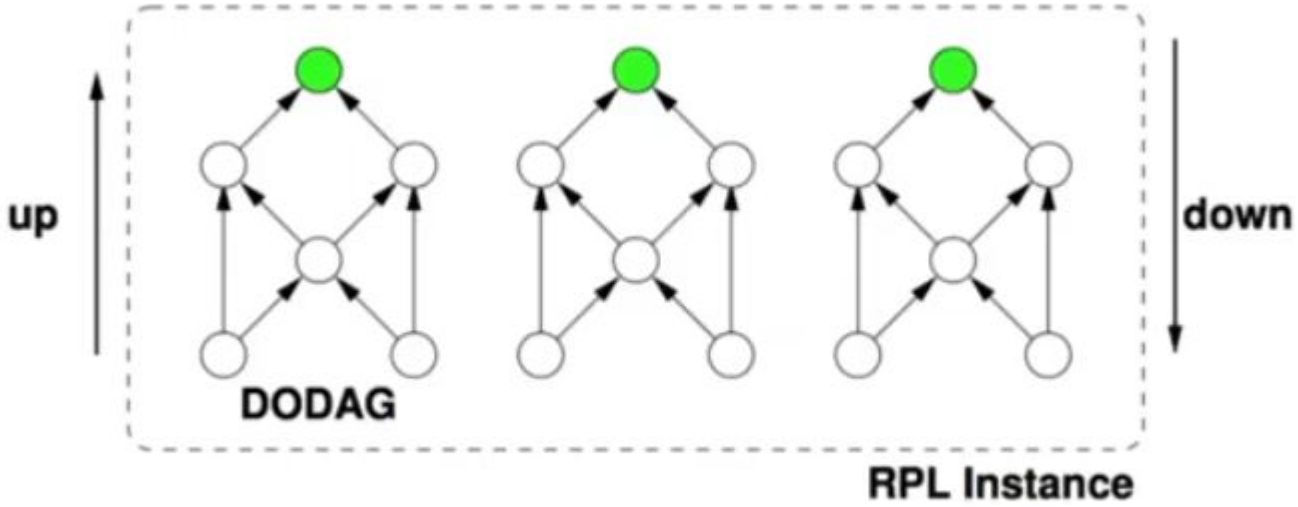
12 traversals

Upward Path

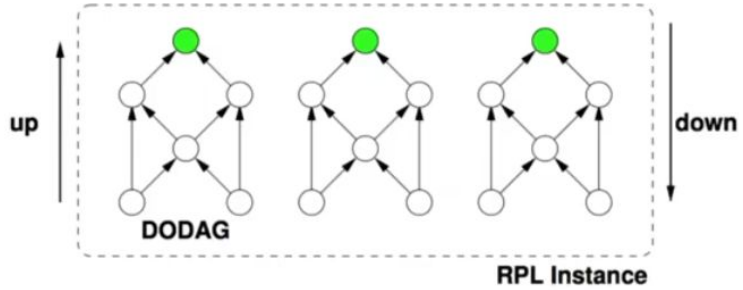


4 traversals

RPL Topology

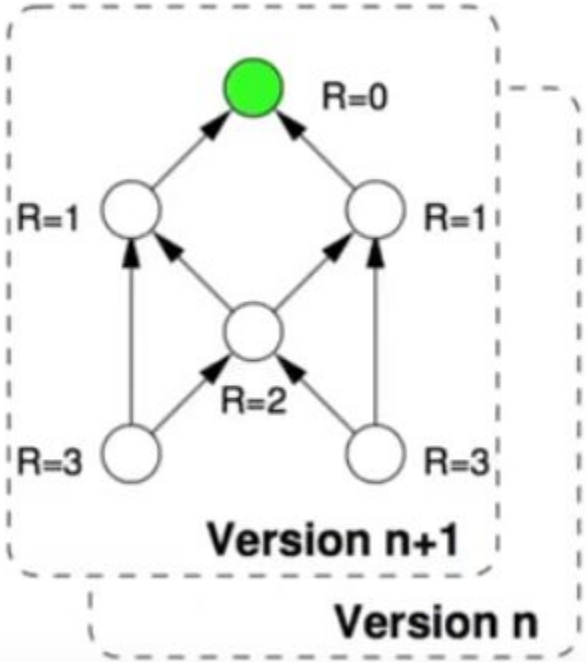


RPL Instance



- DODAGS are disjoint (no shared nodes)
- Link properties: (reliability, latency, . . .) \Node properties: (powered or not, . . .)
- RPL Instance has an optimization objective
- Multiple RPL Instances with different optimization objectives can coexist

RPL Rank



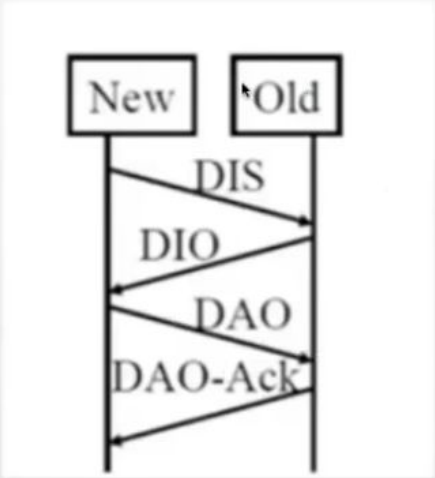
- A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root. The scope of Rank is a DODAG Version.

Forwarding and Routing

- Up routes towards nodes of decreasing rank (parents), Down routes towards nodes of increasing rank
- Nodes inform parents of their presence and reachability to descendants.
- All routes go upwards and/or downwards along a DODAG
- When going up, always forward to lower rank when possible, may forward to sibling if no lower rank exists
- When going down, forward based on down routes

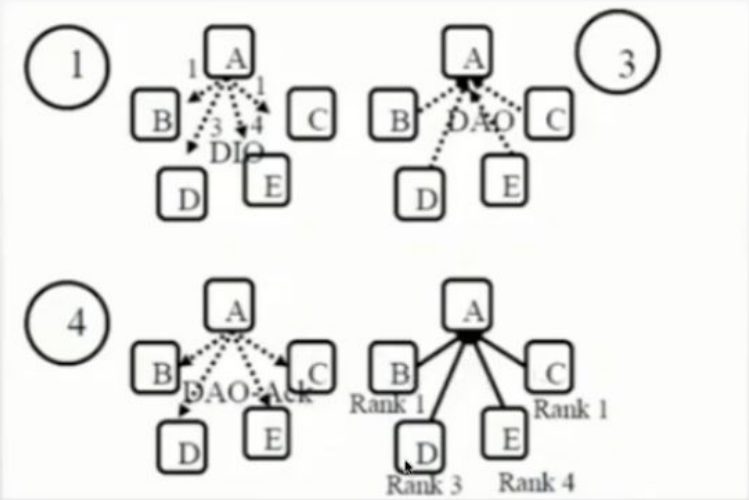
RPL Control Messages

- There are 5 Control Messages, they form the Spanning tree :
- **DODAG information Object (DIO)** : This message is Multicasted downwards . A given node In a DODAG may multicast this message , which lets other nodes know about it , Things like whether the node is grounded or not, whether it storing or non-storing, and it announces other nodes "if they are interested to join , Please Let Me know. "
- **DODAG Information Solicitation (DIS)** : When no announcement is heard, and if a node wants to join a DODAG it sends a control message , For that it wants to know If any DODAG exists. So the message which it sends is Like " Is there any DODAG ? "
- **DODAG advertisement Object (DAO)** : It is A request send by a Child to parent or root. This message requests to allow the child to join to a DODAG.
- **DAO-ACK** : It is a response Send by a root or parent to the child , this response can either be a Yes or No.
- **Consistency check** : deals with security



DODAG Formation

1. Root In a DODAG is a special node. All nodes don't have the capability to be roots. A root is programmed that way.
2. Suppose We have 5 nodes : A, B, C, D and E. And they have to make a DODAG, then following steps are taken:
3. A multicasts DIOs .
4. Rest of the nodes (B,C,D and E) upon receiving DIOs will try to join regardless of their distances. Upon receiving the DIOs they also come to know that their Distance from the A are 1,1,3,4 Respectively.
5. Then B, C, D and E send DAOs to A.
6. A accepts them by replying with DAO -ACK.



DODAG Formation

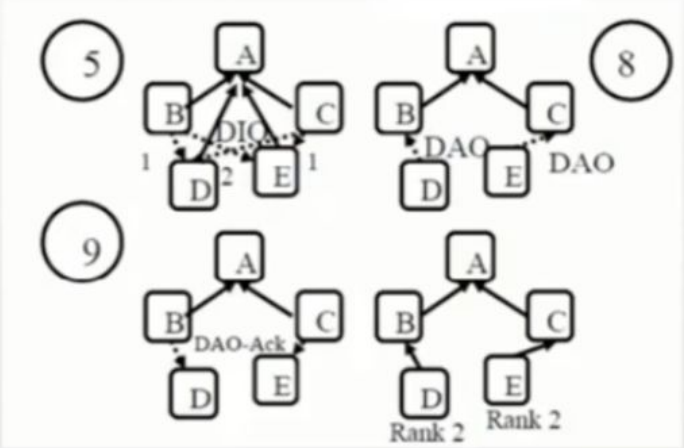
5. Now another round begins, and it starts with the Nodes that are nearest to the root. Since B & C have a rank of 1. They start sending DIOs.

6. D receives those and figures out that its distance from B & C is 1 & 2 Respectively .

7. Similarly E receives these DIOs , and it figures out its distance from B & C to be equal to 2 & 1 respectively .

8. Since D is closer to B and E is closer to C, therefore , D Sends DAO to B and E sends DAO to C.

9. To finalize the DODAG formation , B sends a DAO-ACK to D , and C sends a DAO-ACK to E.



RPL Control messages are ICMPv6 messages

Code: Identify the type of control message

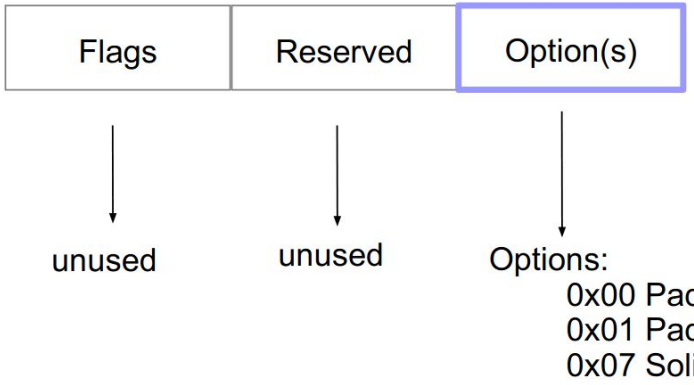
- 0x00 → DODAG Information Solicitation (DIS)
- 0x01 → DODAG Information Object (DIO)
- 0x02 → Destination Advertisement Object (DAO)
- 0x03 → DAO-ACK

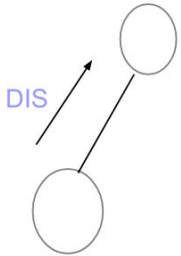
DODAG Information Solicitation (DIS)

Solicit a DODAG Information Object (DIO) from a RPL node

Its use is analogous to that of a Router Solicitation of IPv6 Neighbor Discovery

The DIS Base Object:

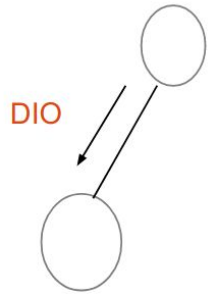




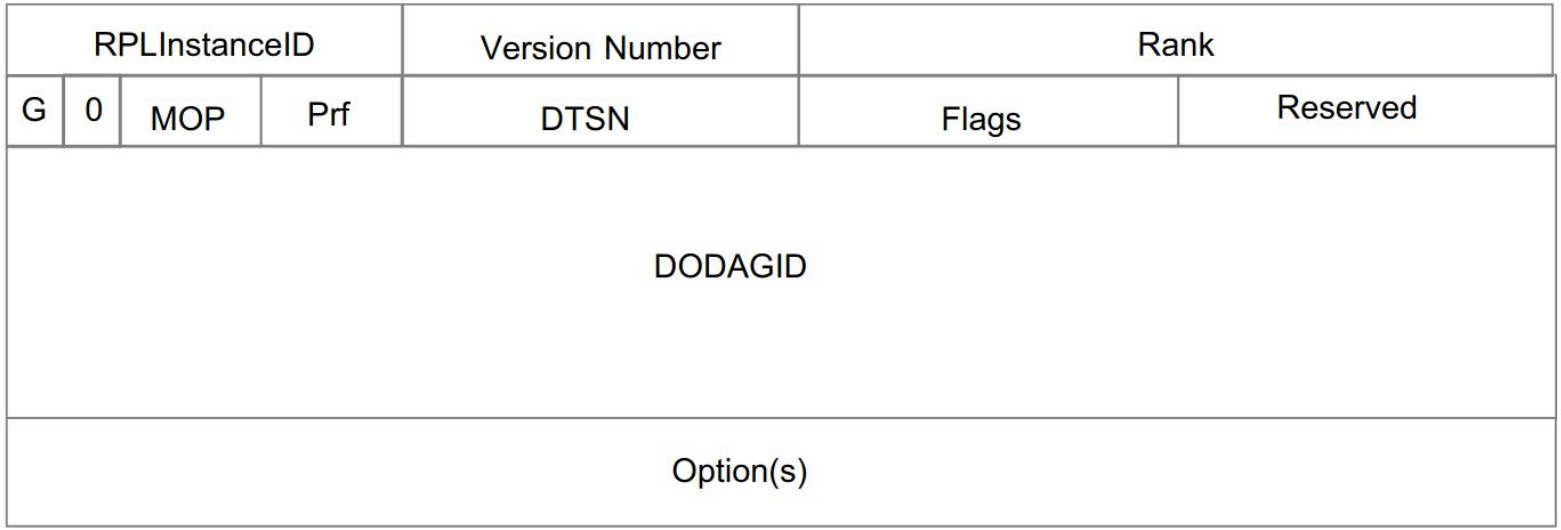
DODAG Information Object (DIO)

Carries information that allows a node to:

- Discover a RPL instance
- Learn its configuration parameters
- Select a DODAG parent set
- Maintain the DODAG



DODAG Information Object (DIO)



DIO Base Object

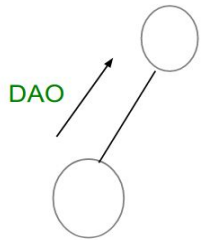
Destination Advertisement Object (DAO)

- Used to propagate destination information Upward along the DODAG.

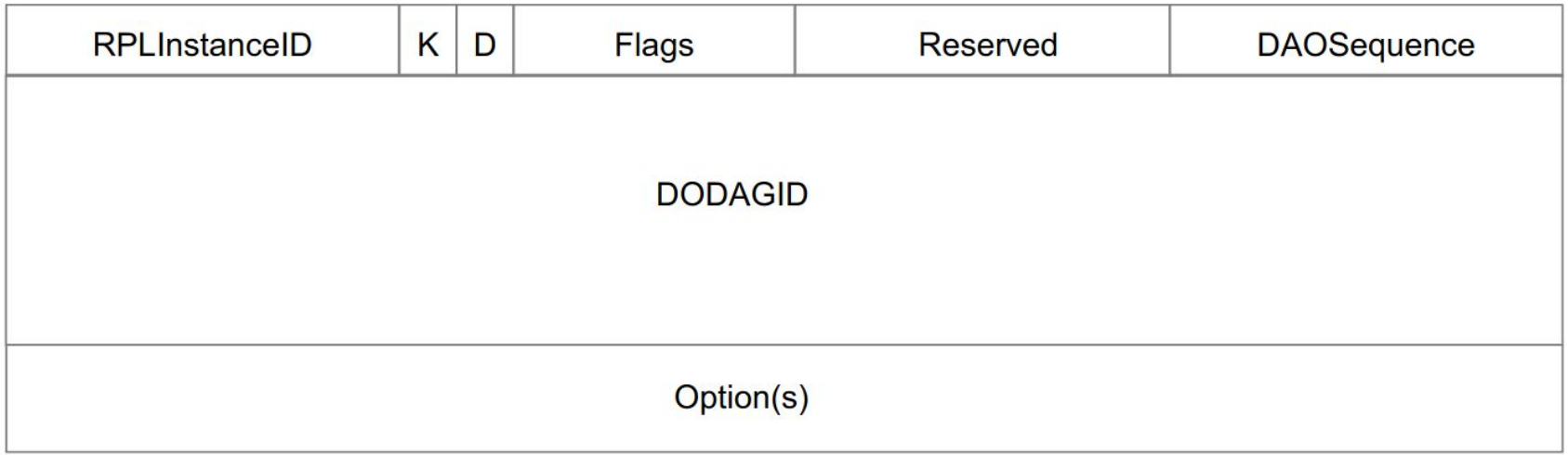
- In **Storing** mode, the DAO message is unicast by the child to the selected parent (s).

- In **Non-Storing mode**, the DAO message is unicast to the DODAG root.

- The DAO message may optionally, upon explicit request or error, be acknowledged by its destination with a Destination Advertisement Acknowledgement (DAO-ACK) message back to the sender of the DAO.



Destination Advertisement Object (DAO)

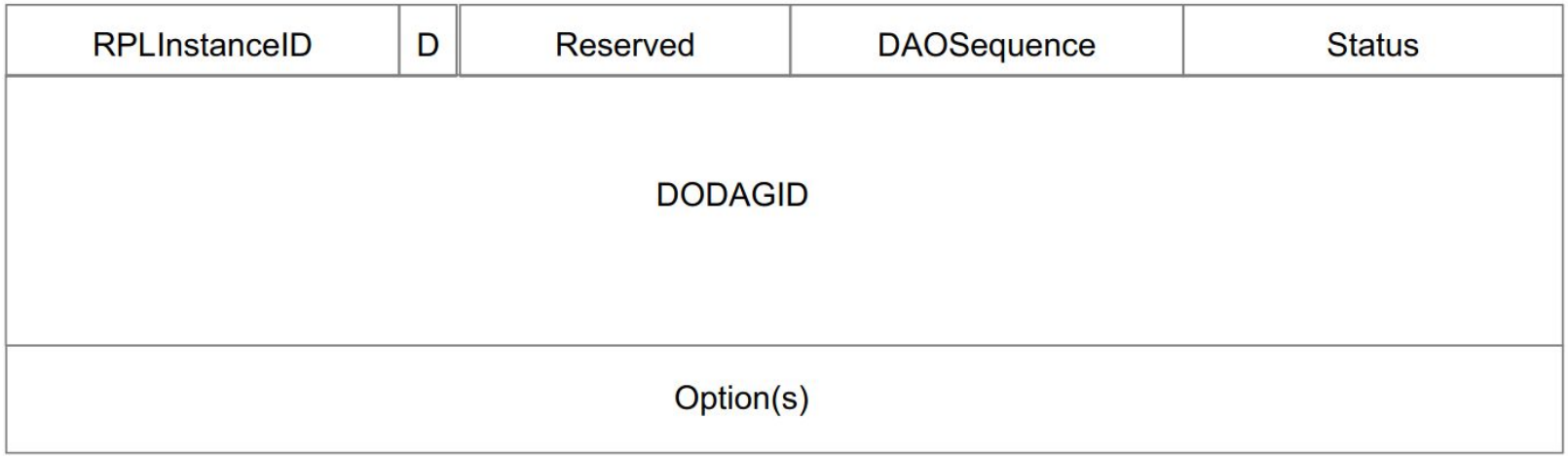


DAO Base Object

* K
  - The ’K’ flag indicates that the recipient is expected to send a DAO-ACK back.
* D
  - The ’D’ flag indicates that the DODAGID field is present.
  - This flag MUST be set when a local RPLInstanceID is used.

- DODAGID (optional)
  - 128-bit unsigned integer set by a DODAG root that uniquely identifies a DODAG.
  - This field is only present when the 'D' flag is set.

Destination Advertisement Object Acknowledgement (DAO-ACK)

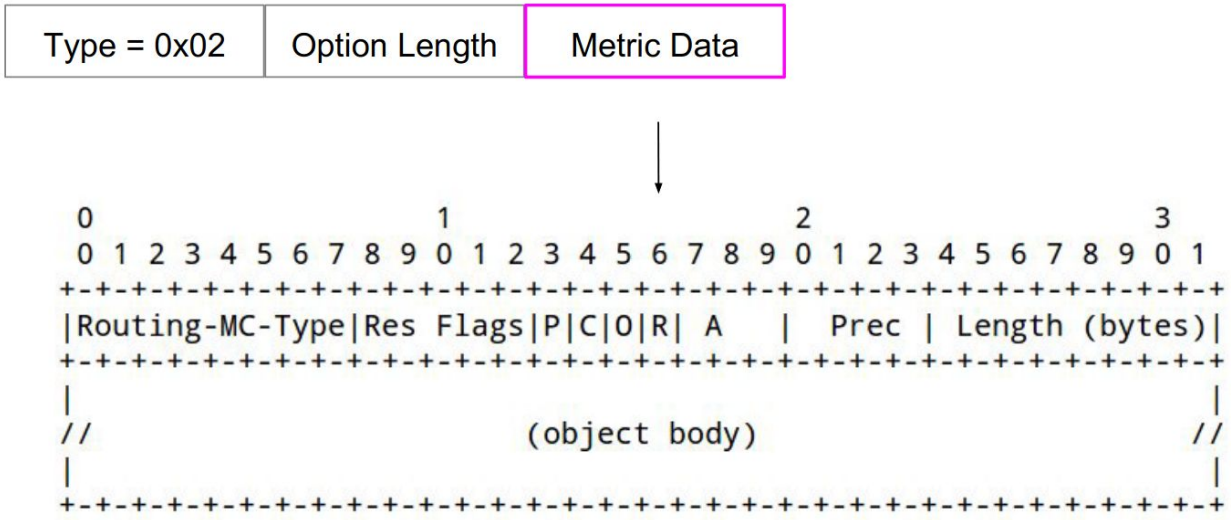


DAO-ACK Base Object

DAG Metric Container

The DAG Metric Container option MAY be present in DIO or DAO messages

The DAG Metric Container is used to report metrics along the DODAG.



Node Metric/Constraint Objects

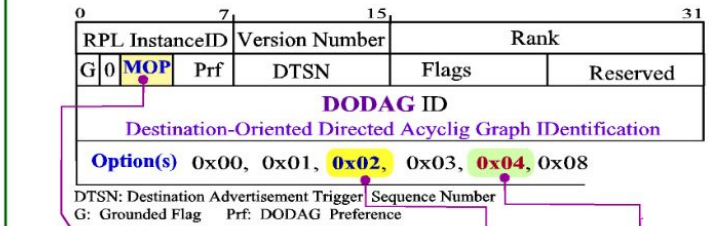
- Node State and Attribute Object
  - Propose to reflect Node workload (CPU, Memory, etc)
- Node Energy Object
  - Constraint
  - three types of power sources: "powered", "battery", and "scavenger"
- Hop Count Object
  - Can be used as metric or constraint
  - Constraint: max number of hops can be traversed
  - Metric: total number of hops traversed

Link Metric/Constraint Objects

- Throughput Object:
  - Currently available throughput (Bytes per second)
- Latency:
  - Can be used as a metric or constraint
  - Constraint: Max latency allowable on path
  - Metric: aditive metric updated along path
- Link Reability:
  - Link Quality Level Reliability (LQL): 0=Unknown, 1=High, 2=Medium, 3=Low
  - Expected Transmission Count (ETX) (Average number of TX to deliver a packet)
- Link Colour:
  - Metric or constraint, arbitrary admin value

RPL message DIO (DODAG Information Object)

for DODAG setup or modification



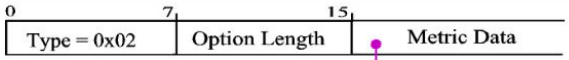
MOP Mode of Operation

0 No Downward Routes
1 Non-Storing MOP
2 Storing MOP, no Multicast Support
3 Storing MOP, Multicast Support

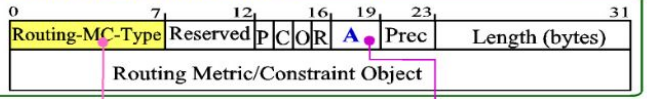
DAG Metric Container with the **Routing Metric/Constraint** Specification

DODAG Configuration with the **Objective Function** Type Specification

0x02: DAG Metric Container



DAG Metric Container Format RFC 6551



Routing-Matric/Constraint -Type

0 Unassigned
1 Node State and Attribute
2 Node Energy (NE)
3 Hop Count (HC)
4 Link Throuput
5 Link Latency
6 Link Quality Level (LQL)
7 Link ETX
8 Link Color

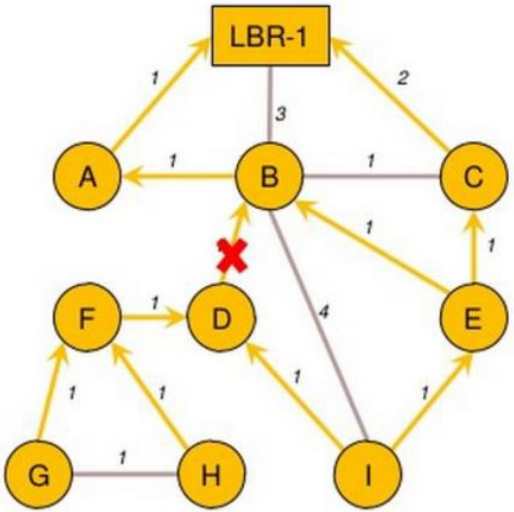
RM: Routing Metric

A = 0 RM is additive
A = 1 RM reports a maximum
A = 2 RM reports a minimum
A = 3 RM is multiplicative

Aggregated RM Characteristics

Anatol Badach ResearchGate

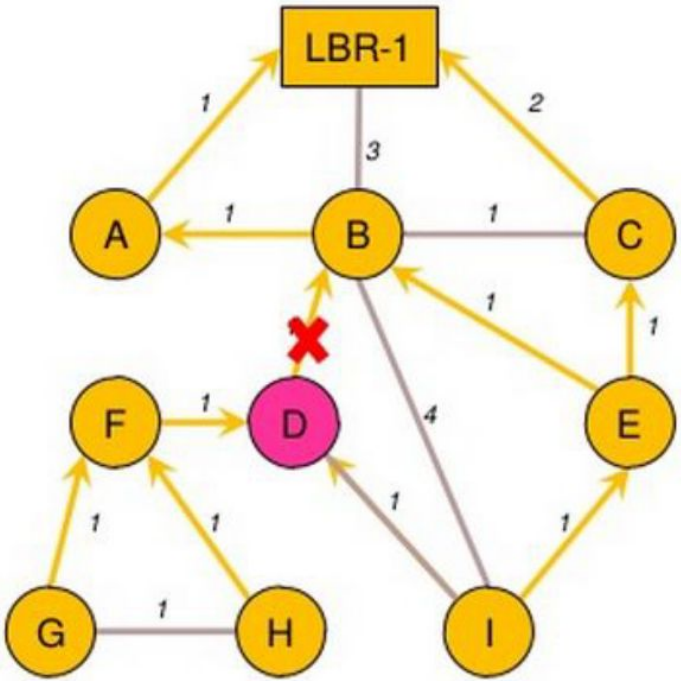
RPL has mechanism for loop detection and DODAG Repair



Suppose link between nodes B and D is broken: (for instance, a metal door closes!)

- Node D type node B in its list
- Parent Node D is no longer any time in grounded DODAG Parent, so it will be the root of floating DAG itself

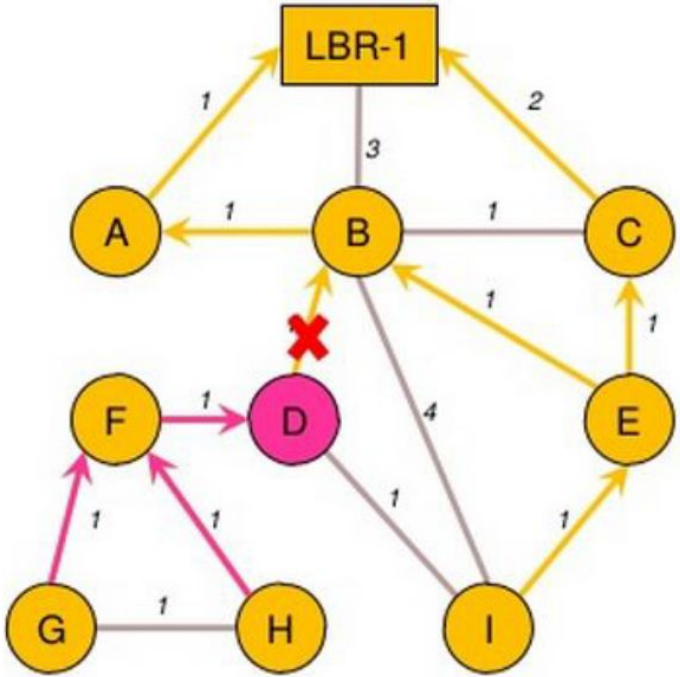
RPL has mechanism for loop detection and DODAG Repair



Node D play DIO to notify change of sub-DAG

- Node I has alternate parent E, so it does not leave the DAG of LBR-1
- I eliminates Node D Node from possible Parents list

RPL has mechanism for loop detection and DODAG Repair

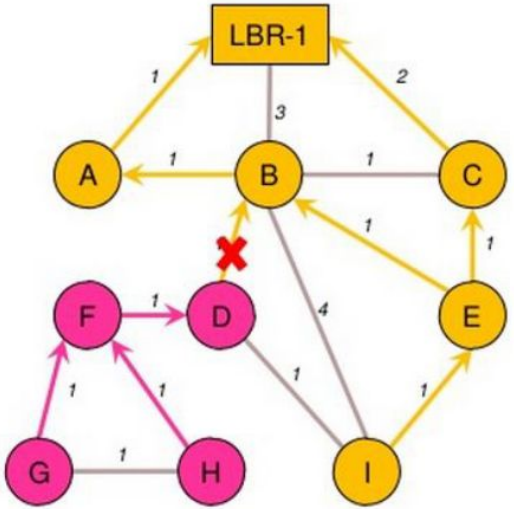


- Node F follows D node, as D leaves LBR-1 DAG: it has no choice

- Node F hears DIO from D.

- Node G and H also follow floating node F DAG

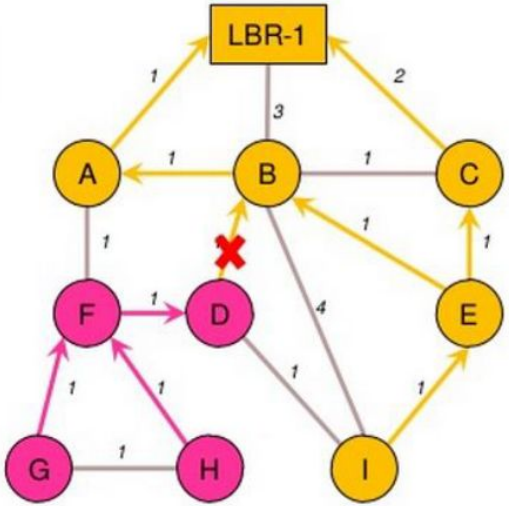
RPL has mechanism for loop detection and DODAG Repair



Node I found DIO

- Node D listens to DIOs for opportunities to re-enter the last Grounded with depth 5 Node I

RPL has mechanism for loop detection and DODAG Repair

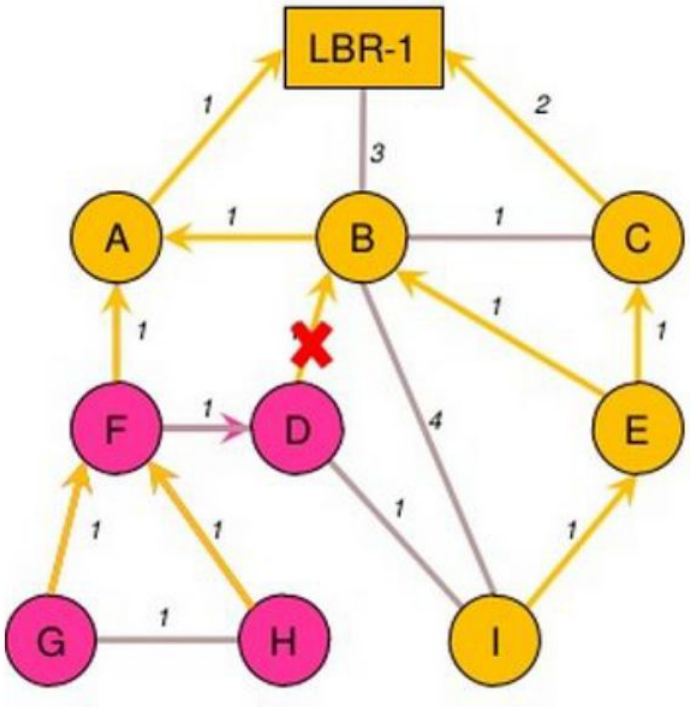


- Suppose link between A and F is set
- Node A send DIO

- Node F release notice Grounded DAG re-entry opportunities with depth 2 through node A DAG

- Hop Node F started with 1 cycle timer associated with the node A

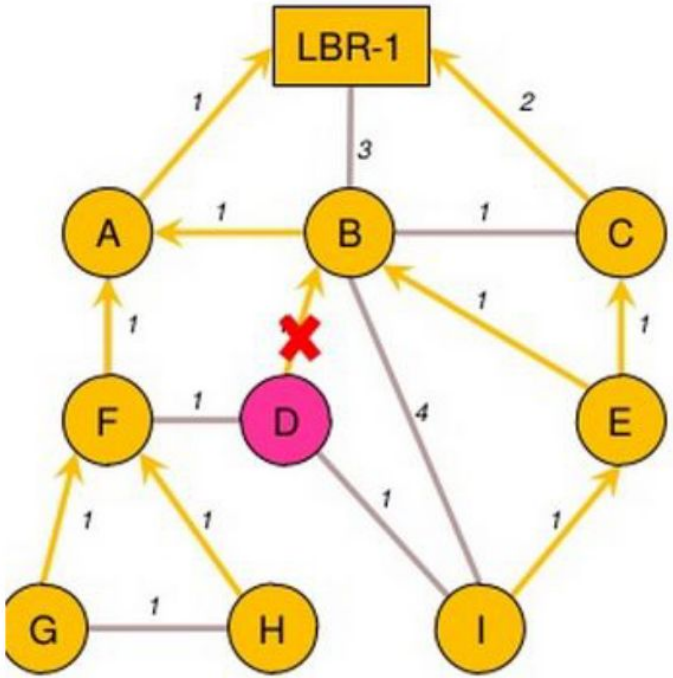
RPL has mechanism for loop detection and DODAG Repair



- DAG node F Timer goes off, issues DIS, receives DIO from A!

- Node F Grounded DAG with depth 2 by adding the Parent A
- Node F send DIO with new rank/etc.
- Node G and H join to the Grounded DODAG through F

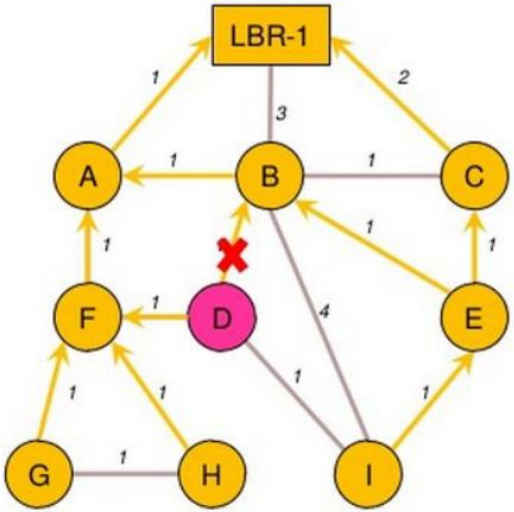
RPL has mechanism for loop detection and DODAG Repair



Node D hears new DIO from F.

Node D start DAG Hop cycle timer with 2 attached to node F, while other timer is running DAG Hop with 4 cycles associated with the first node

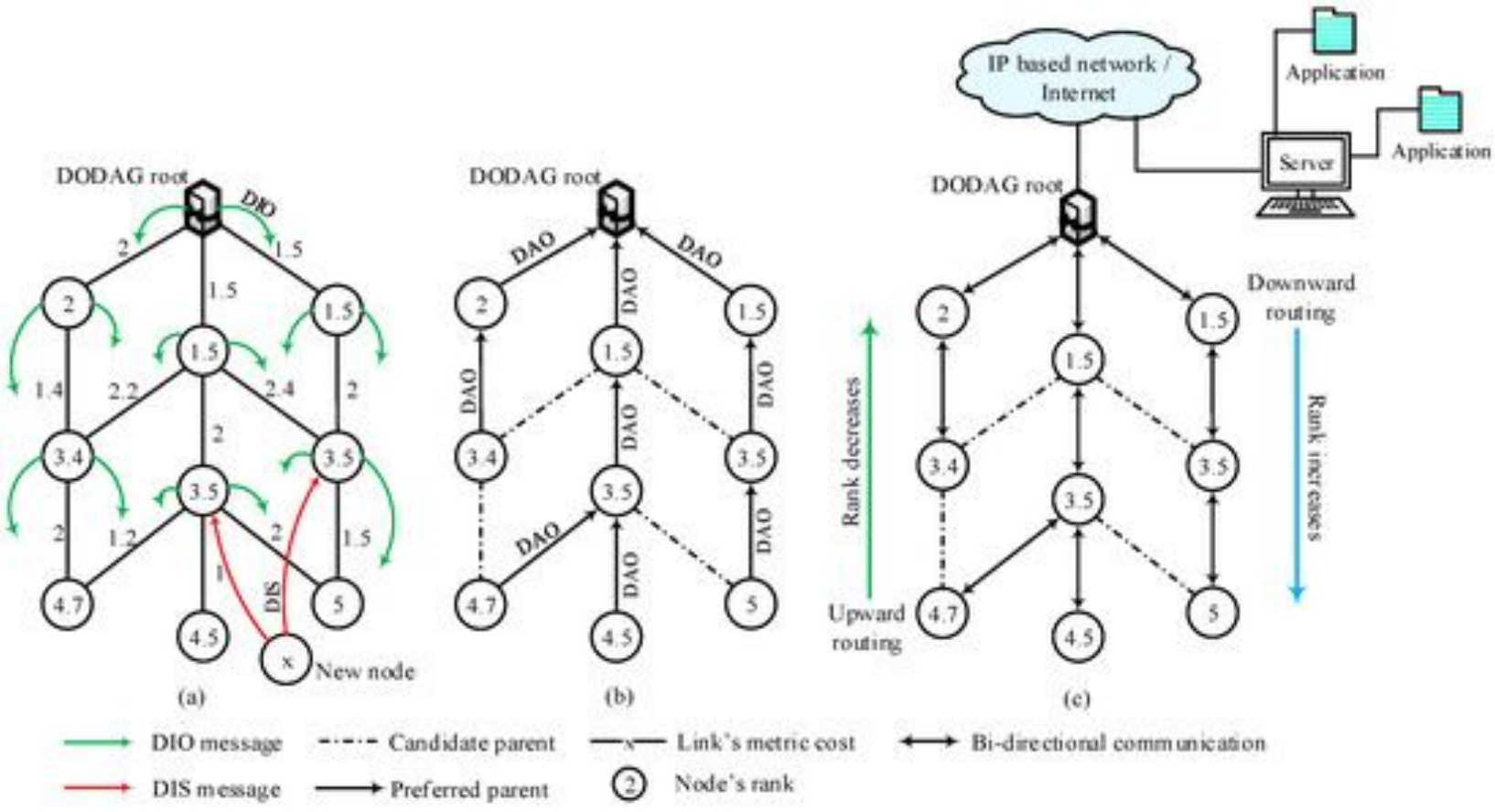
RPL has mechanism for loop detection and DODAG Repair

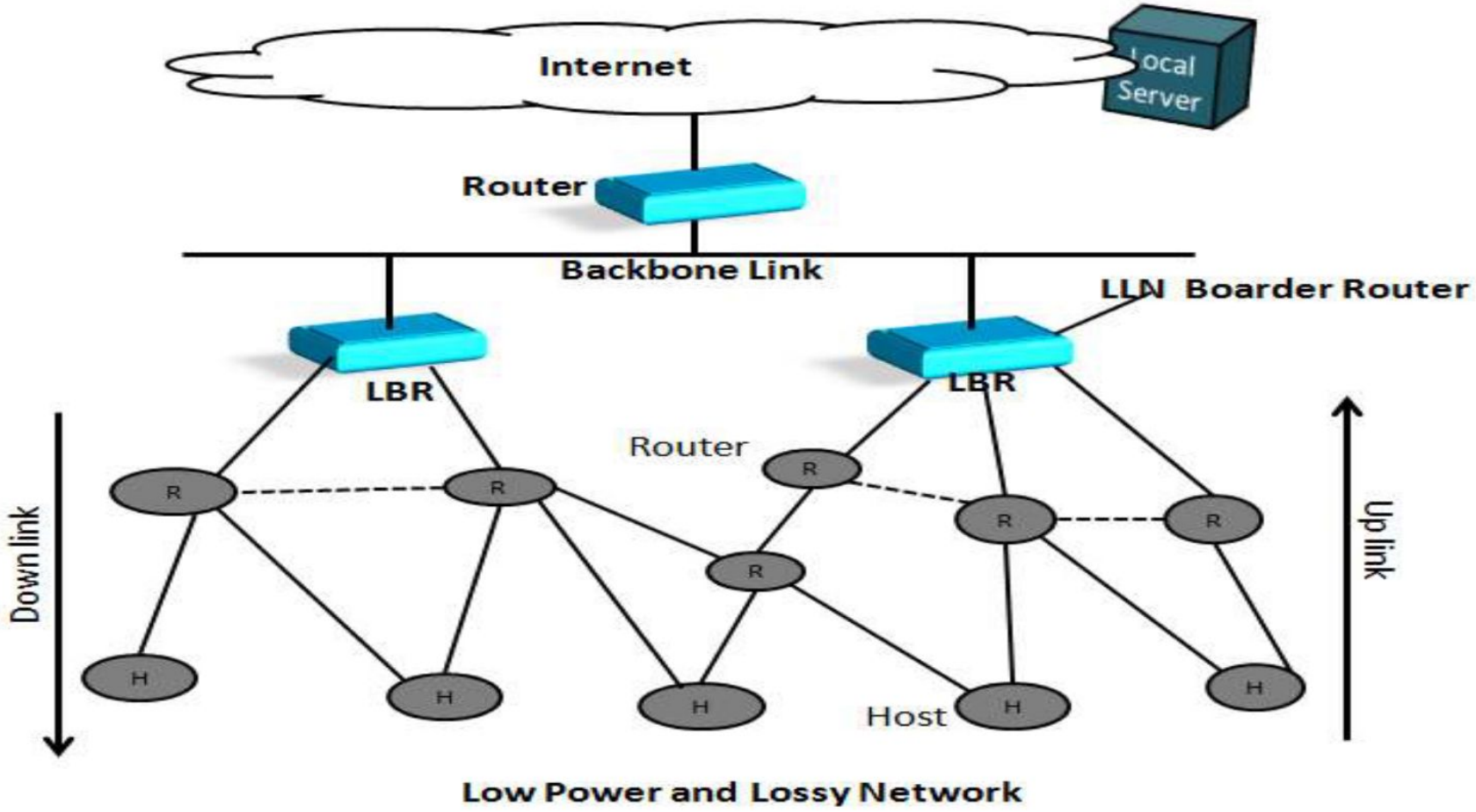


- DAG node D Hop timer with 2 cycles tend to end first.
- Node D engaged with depth 3 Grounded

DAG by adding Node F's parent.

- End


DODAG root
DIO
2
1.5
1.5
2
1.5
1.4
2.2
2.4
2
2
3.4
3.5
3.5
2
1.2
2
1.5
1
DIS
4.7
4.5
5
x
New node
(a)
DODAG root
DAO
DAO
2
1.5
DAO
DAO
DAO
1.5
DAO
3.4
3.5
DAO
3.5
DAO
DAO
4.7
4.5
5
(b)
IP based network / Internet
Application
Application
Server
DODAG root
Downward routing
2
1.5
1.5
Rank decreases
Rank increases
3.4
3.5
3.5
Upward routing
4.7
4.5
5
(c)
DIO message
Candidate parent
Link's metric cost
Bi-directional communication
DIS message
Preferred parent
2
Node's rank


Internet
Local Server
Router
Backbone Link
LLN Boarder Router
LBR
LBR
Router
R
R
R
R
R
R
H
H
H
H
H
Host
Down link
Up link
Low Power and Lossy Network

References

* T. Winter, et al., “RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks,” Internet Engineering Task Force (IETF), RFC 6550, March 2012.