SAPIENZA
UNIVERSITÀ DI ROMA

RPL: Routing for IoT

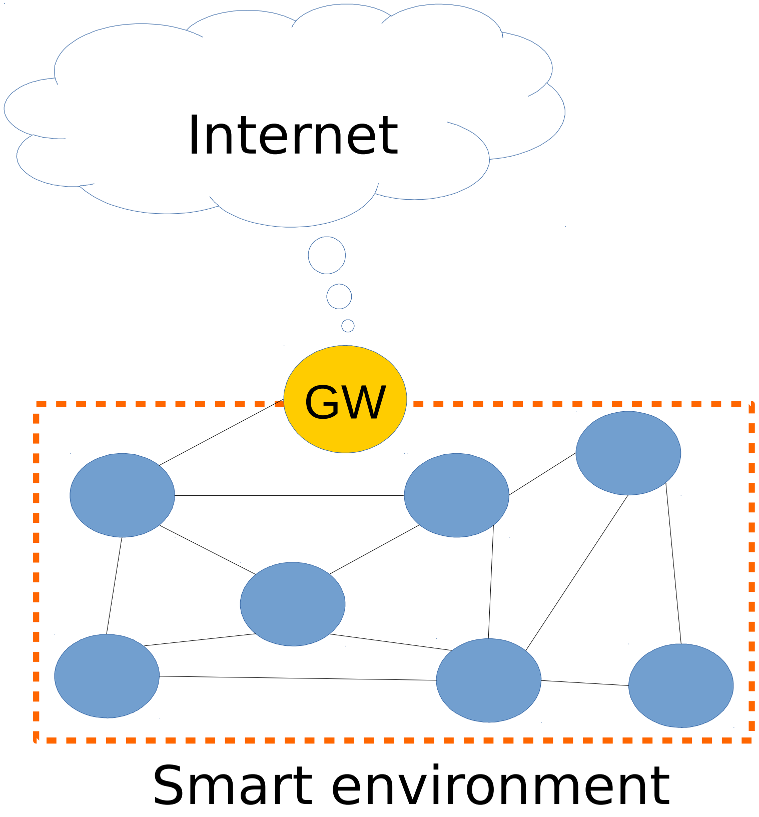
Bardh Prenkaj
Dept. of Computer Science

Overview

- Protocol scenario description
- Design principles of the protocol
- Fundamental terminology to understand RPL
- Importance of the objective function
- *DODAG* Versioning
- “RPL in the wild”
- Forwarding data on IoT networks
- Brief introduction to message timing: the *Trickle* algorithm

Scenario description

- **Components**: gateway (GW), smart devices (blue circles) and the Internet (cloudy shape)
- GW **connects** the smart devices (SDs) with Internet
- SDs want to **communicate** with the **Internet**
- Need of a **protocol** that **routes** packets from SDs to GW and then to the Internet



LLNs

- Low-Power and Lossy Networks
- Routers **constrained** in processing **power**, memory and **energy**
- Links characterized by high **loss** rates, low data rates and **instability**
- Traffic flows include
  - **point-to-point**
  - **point-to-multipoint**
  - **multipoint-to-point**

IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL)

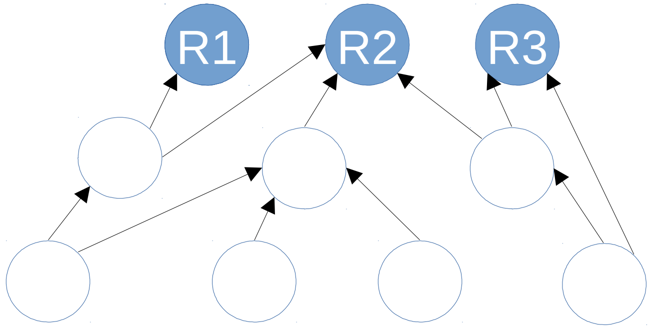
Design Principles

- Provides a mechanism for **multipoint-to-point** and a **point-to-multipoint** traffic[1]
- Designed to **meet** the requirements in RFC5867, RFC5826, RFC5673 and RFC5548
- **Separate** packet **processing** and **forwarding** from the **routing** optimization objective
- Router **reachability** must be verified before the router can be used as a parent (i.e.: Neighbour Unreachability Detection - **NUD**)

[1] Support for point-to-point is also available

Terminology

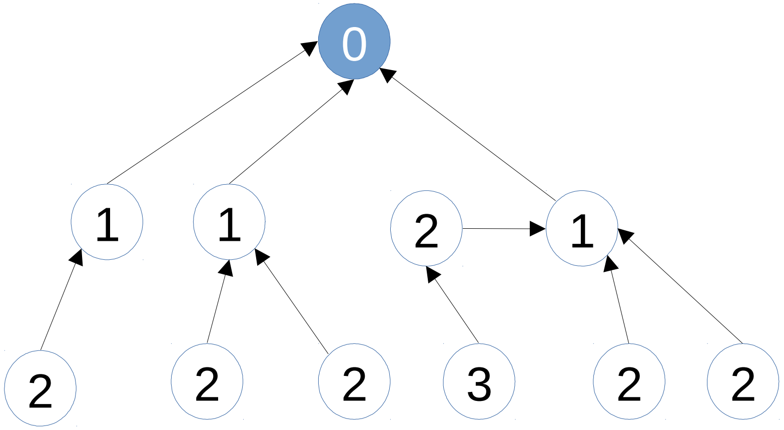
- Directed Acyclic Graph (**DAG**) – a directed graph without cycles
- DAG **root** – node within the DAG that has no outgoing edge



- Destination-Oriented DAG (**DODAG**) – a DAG with a single root

Terminology

- **Up** – the direction from leaf nodes towards DODAG roots following DODAG edges
- **Down** – the direction from DODAG roots towards leaf nodes in the reverse direction of DODAG edges
- **Rank** – defines the node's individual position relative to other nodes w.r.t. a DODAG root. Rank's computation depends on the DAG's **objective function**



Node ranking based on distance from the root

Terminology

- **RPL Instance ID** – a unique identifier within a network
- **RPL Instance** – a set of one or more DODAGs that share a RPL instance ID. Each RPL instance operates independently of other instances
- **Goal** – an application-specific goal that is defined outside the scope of RPL. Any node that roots a DODAG will need to know about this Goal to decide whether or not the Goal can be satisfied

Terminology

- **DODAG ID**[2] – a unique identifier of a DODAG root. The tuple **(RPL Instance ID, DODAG ID)** uniquely **identifies** a DODAG
- **Grounded DODAG** – when the DODAG root can satisfy the goal
- **Floating DODAG** – a DODAG is called floating if it's not grounded
- **DODAG parent** – immediate successor of a node in the path towards the root

[2] A DODAG ID is an IPv6 address

Terminology

- **Storing DODAG** – each node in the DODAG keeps routing tables (no global view of the network)
- **Not-Storing DODAG** – nodes don’t have information of routing besides their corresponding parents

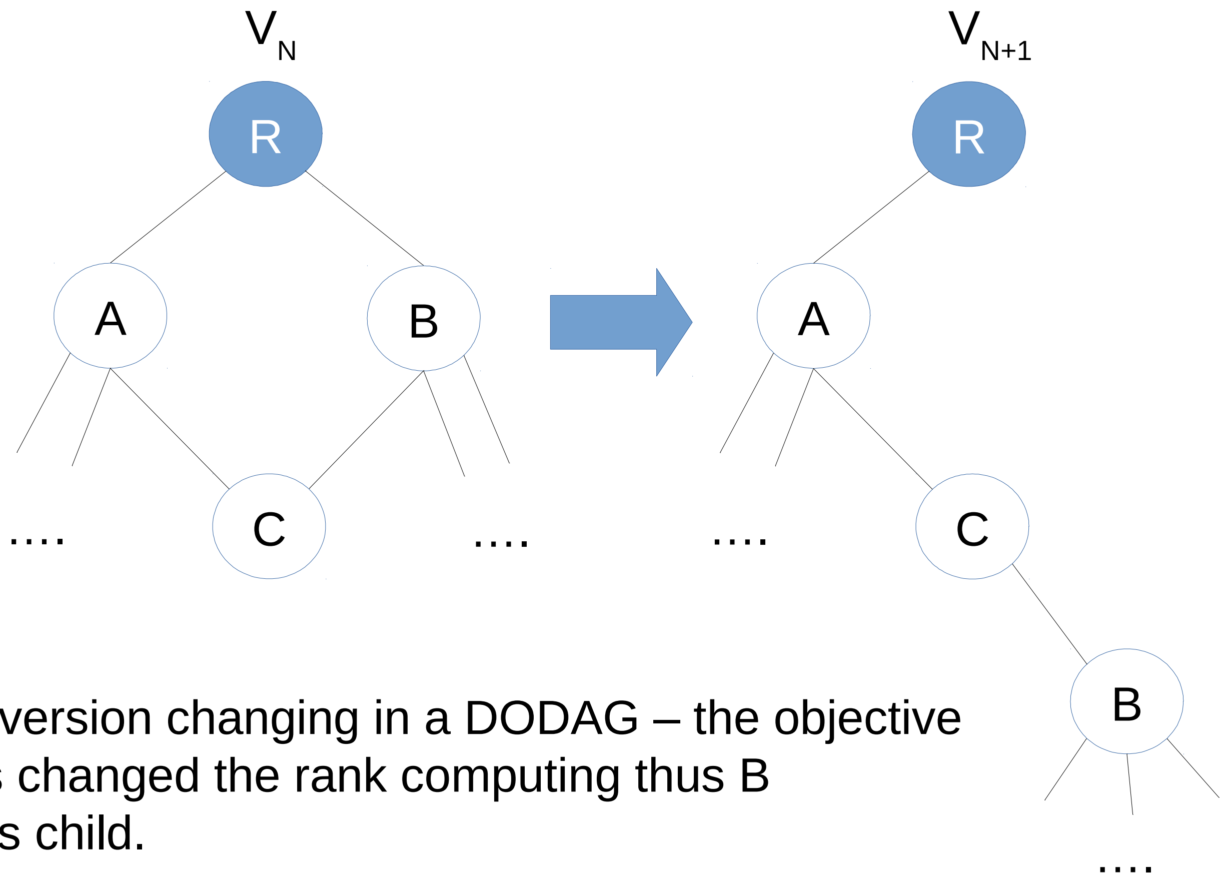
Objective Function Importance

- Defines the **node ranking computation** knowing routing metrics and optimization objectives
- **Dictates parent selection** in a DODAG
- Implies the **construction** of the DODAG
- Allows to **minimize** some **constraints** (i.e..: power consumption) given specific metrics

DODAG Versioning

- Every time a **new** DODAG gets computed with the same root, the **version changes**
- Change of versions: **broken** links, **joining** nodes, **failing** nodes
- The DODAG's **lifetime** passes through different possible versions
- The **same** objective function on two different DODAGs can have **different versions**

DODAG Versioning



Example of version changing in a DODAG – the objective function has changed the rank computing thus B becomes C’s child.

RPL: Message Types

- **DIO** (DODAG Information Object) – it is sent in **multicast downwards**
  - The **root** can send the DODAG’s version, objective function, its ID to the other nodes in the network
- **DIS** (DODAG Information Selection) – it may be used to solicit a DODAG Information Object from an RPL node
  - A node wanting to join the DODAG, sends in **broadcast** this message (neighbourhood discovery)

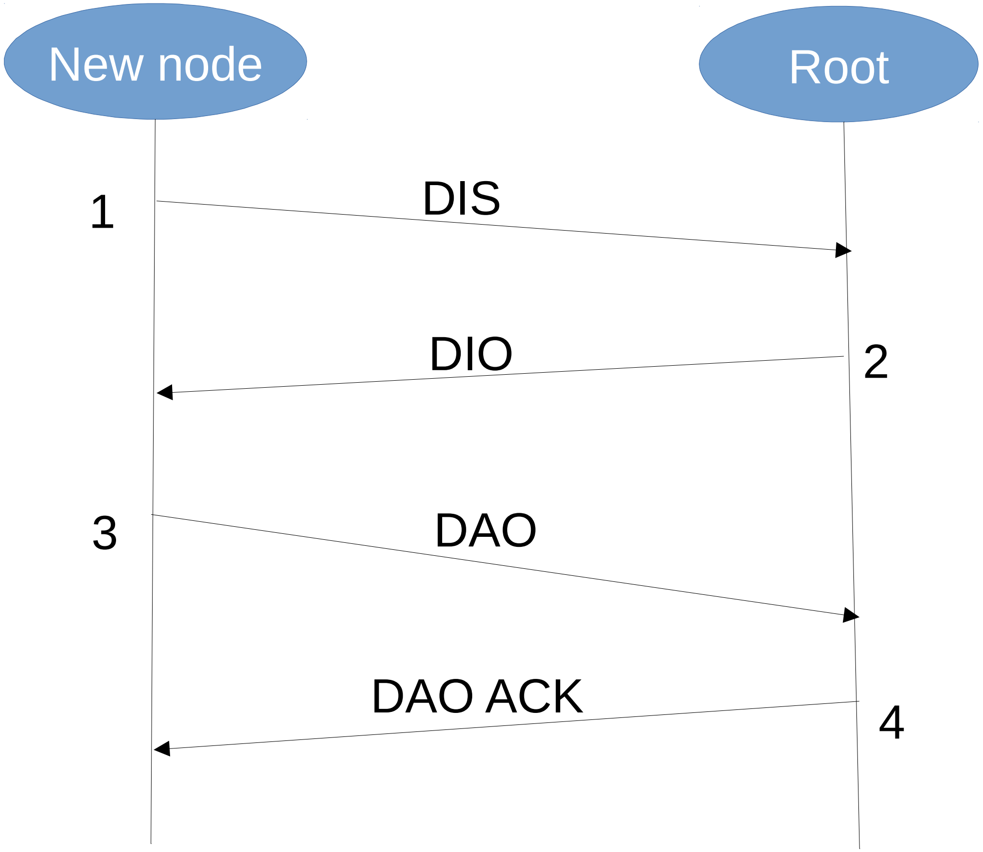
RPL: Message Types

- **DIO** (DODAG Information Object) – it is sent in **multicast downwards**
  - The **root** can send the DODAG's version, objective function, its ID to the other nodes in the network
- **DIS** (DODAG Information Selection) – it may be used to solicit a DODAG Information Object from an RPL node
  - A node wanting to join the DODAG, sends in **broadcast** this message (neighbourhood discovery)

RPL: Message Types

- **DAO** (DODAG Advertisement Object) – it is sent **upwards** towards the root
  - Usually its usage is limited into **answering** to a DIO message (*“Can I join you?”* message)
- **DAO ACK** – it is a **response** to a DAO message with an ACK or NACK
  - Acknowledge that a **new** node has **joined** an existing DODAG

Node message exchanging



How does RPL work?

- **Goal**: construct a DODAG satisfying an **objective function**
- **Assumption**: some nodes are **configured** to be **roots** (however only one is chosen because of DODAGs properties)

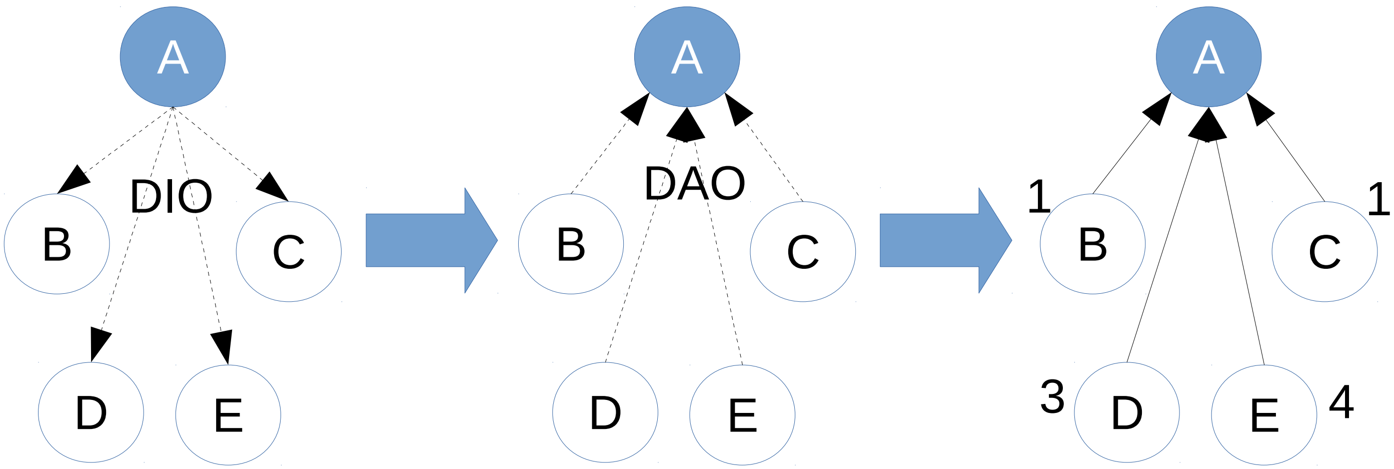
How does RPL work?

- **Root** sends **DIO** messages to the other nodes in its **transmission range**
- Each node, upon receiving a DIO, use their **own information** to **decide** whether to reply or not to the root
- The replying nodes provision **routing table entries** accordingly
- The replying nodes update their corresponding ranks according to the objective function received in the DIO message

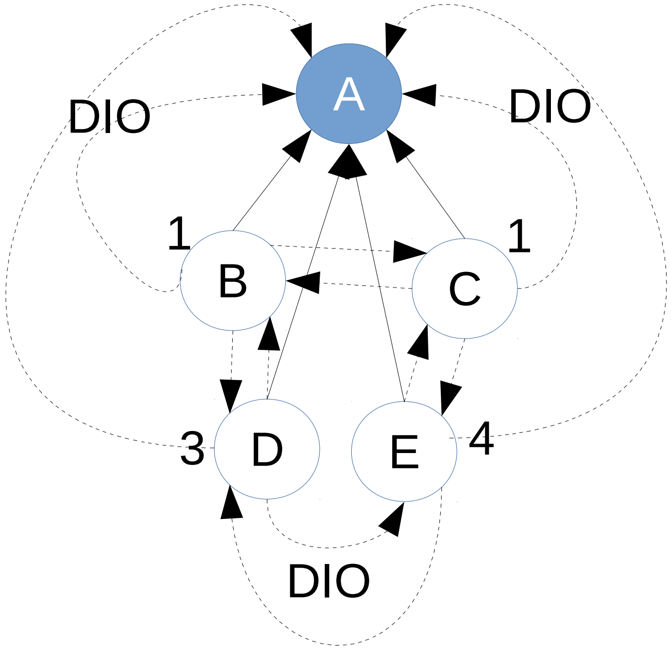
How does RPL work?

1) A sends a DIO message in **multicast**
2) B, C, D and E respond with a DAO in **unicast**
3) A sends a DAO ACK to each of the replying nodes in **unicast**
4) The process **iterates** for the new attached nodes to the constructed DODAG
5) If a nodes receives a **lower** rank advertisement, it chooses that rank and the advertising node as its **parent**

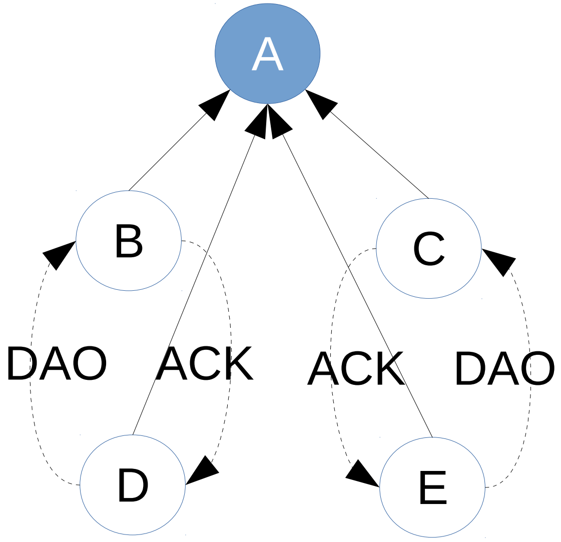
How does RPL work?



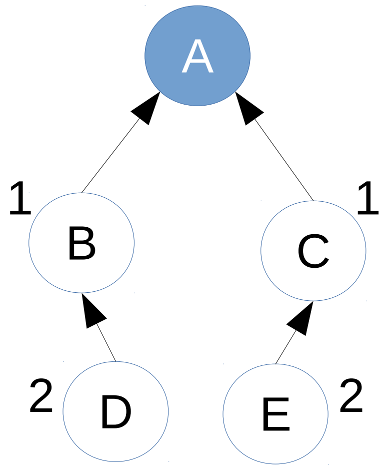
How does RPL work?



Suppose C advertises to E rank 1 and B advertises 1 and 2 to D and E, respectively



E accepts C as a parent and D accepts B as parent



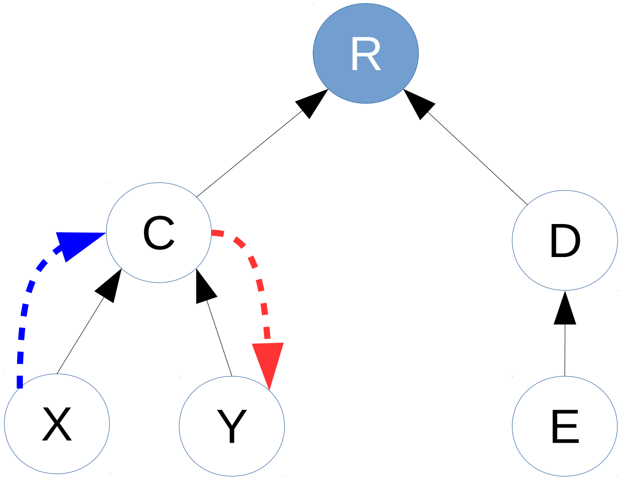
The new DODAG is constructed and the nodes update their ranks

Forwarding data in LLNs

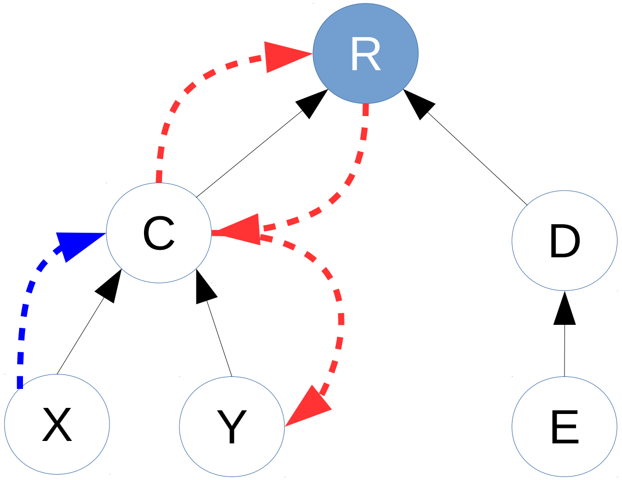
- **1-to-1**: (from X to Y)
  - **Storing DODAG →** X sends data to its chosen parent and the latter, since it has the IDs of its subtree children, forwards the packet to Y
  - **Not-storing DODAG →** X's parent has to forward the packet to the root who then provides to forward the packet to Y

Forwarding data in LLNs

- **1-to-1**: (from x to y)



Storing DODAG case: X sends to C (blue) and C forwards data to Y (red)



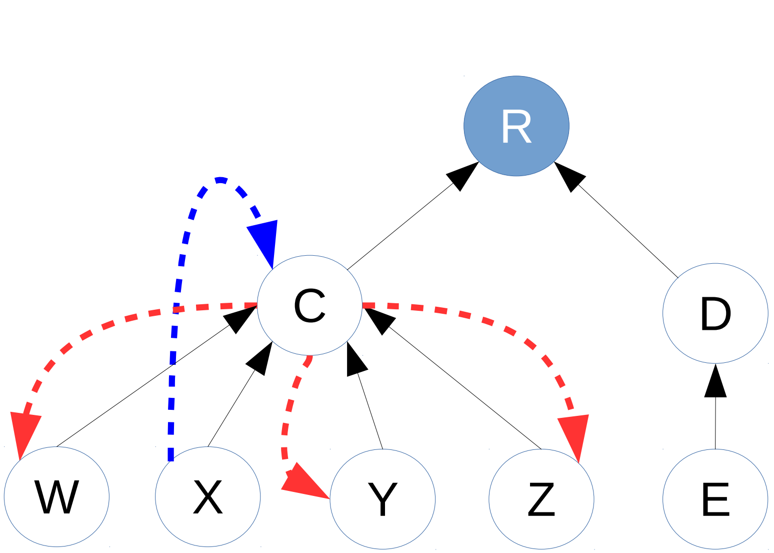
Not-storing DODAG case: X sends to C (blue), C forwards data to the root R who then forwards the data to Y by encapsulating the entire path in the packet (red)

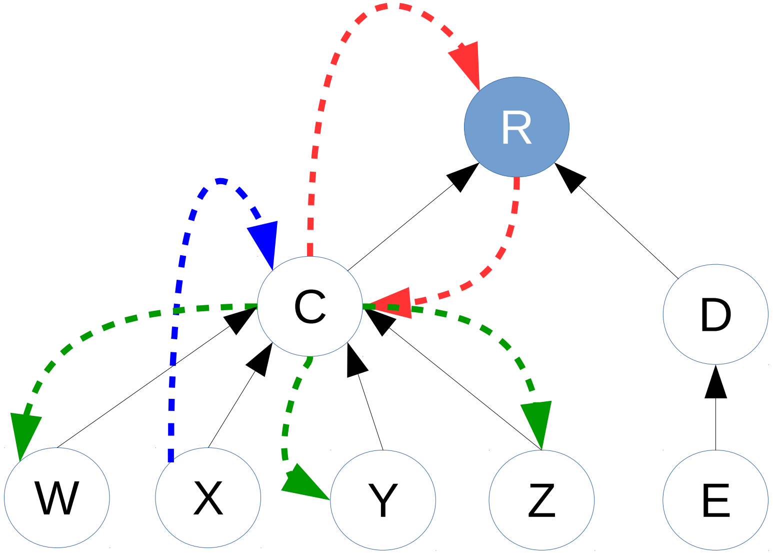
Forwarding data in LLNs

- **1-to-n**:
  - **Storing DODAG** → each node broadcasts data packets to its children
  - **Not-storing DODAG** → the root has to encapsulate the whole path to the destination and forward the packet

Forwarding data in LLNs

- **1-to-n**:





Storing DODAG case: X sends to C (blue) and C broadcasts to its children different from the source X (red)

Not-storing DODAG case: X sends to C (blue), C forwards data to the root R (red) who then forwards the data to W,Y,Z by encapsulating the entire path to these nodes. C unwraps the received packet and forwards it to the destinations (green)

Message timing: The Trickle Algorithm

- Nodes transmit when an **alarm** is solicited after some interval of time
- **Regulates** DIO broadcast transmissions
- If the network is **consistent**, then nodes do **not** transmit **redundant** messages
- If **inconsistency** is detected, then the transmitting timer is **decreased**; otherwise the timer is increased

Further readings

- RFC 6550 @ https://tools.ietf.org/html/rfc6550
- Iova Oana, Pietro Picco, Timofei Istomin, and Csaba Kiraly. "RPL: The Routing Standard for the Internet of Things... Or Is It?." IEEE Communications Magazine 54, no. 12 (2016): 16-22.
- https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/rpl/configuration/15-mt/rpl-15-mt-book.html
- IPSO Alliance. RPL: The IP routing protocol designed for low power and lossy networks @ http://www.ipso-alliance.org/wp-content/media/rpl.pdf
- Aishwarya Parasuram, David Culler and Randy Katz. “An Analysis of the RPL Routing Standard for Low Power and Lossy Networks”. (2016) @ https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-106.pdf