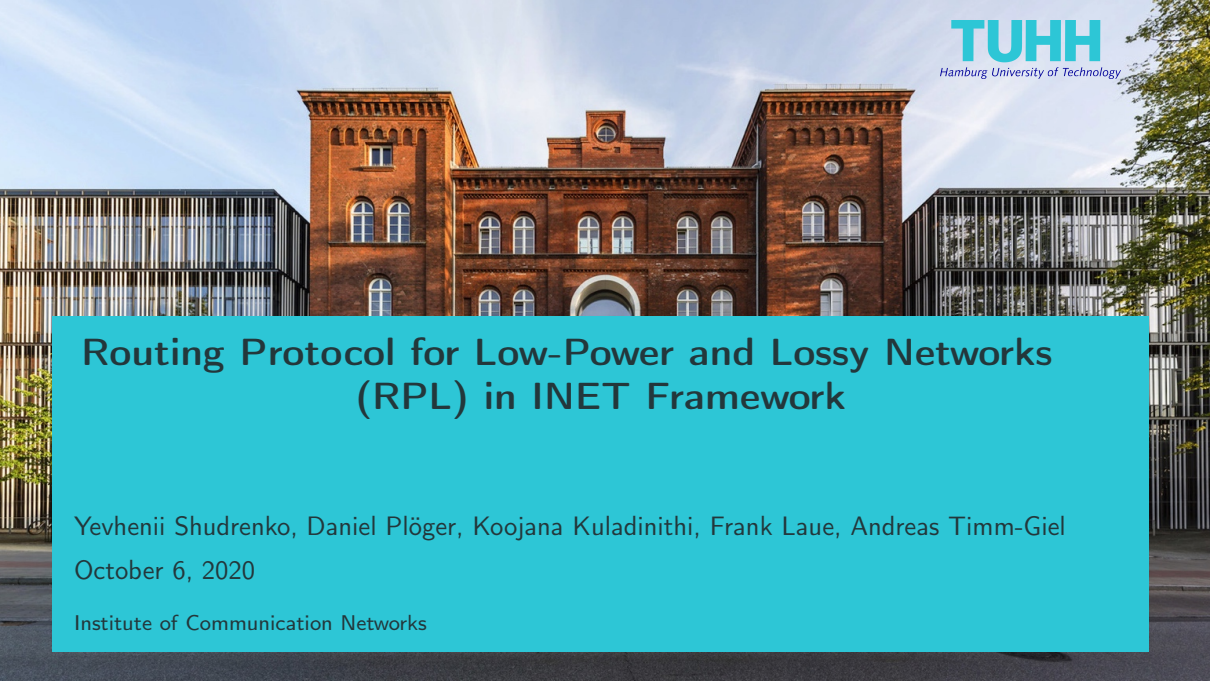


TUHH
Hamburg University of Technology

Routing Protocol for Low-Power and Lossy Networks (RPL) in INET Framework

Yevhenii Shudrenko, Daniel Plöger, Koojana Kuladinithi, Frank Laue, Andreas Timm-Giel

October 6, 2020

Institute of Communication Networks

Introduction

Proactive distance-vector routing protocol for Low-Power and Lossy Networks (LLNs).

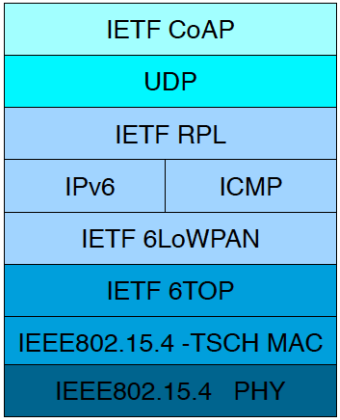


Figure 1: RPL in IETF standardized stack[1]

Features

- IPv6 compatible & Standardized [RFC 6550]
- Scalable
- Point-to-point (P2P), point-to-multipoint (P2MP), multipoint-to-point (MP2P) communication modes
- Configurable objective function per application requirements
- Storing, non-storing modes to match memory constraints
- *Trickle* timer [RFC 6206] enables flexible reactivity behaviour
- DAG inconsistency detection and repair

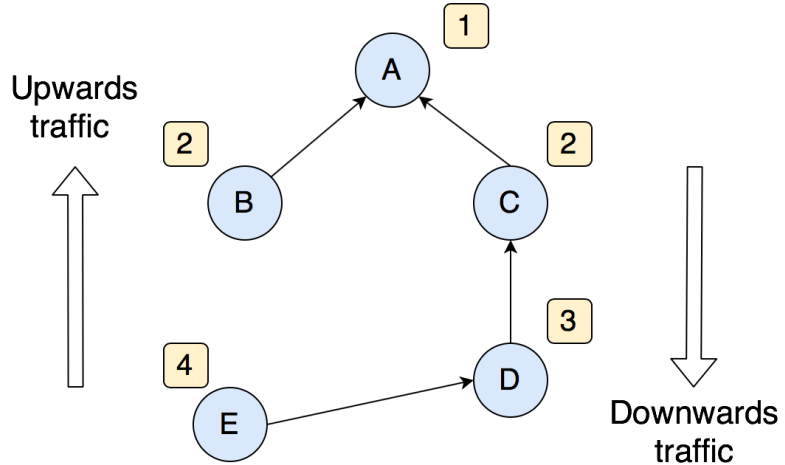


Figure 2: Destination Oriented Directed Acyclic Graph (DODAG)

Defines criteria for preferred parent selection

Table 1: Some of commonly used objective function metrics [2]

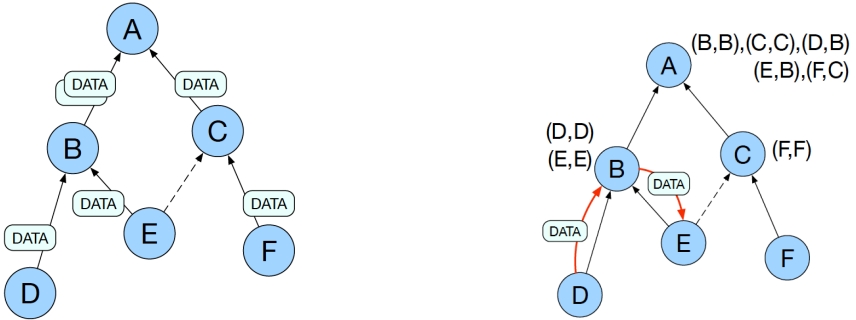
Node Metric	Link Metric
State: CPU, Memory Load etc.	Throughput
Energy	Latency
Hop Count	Reliability

Control Messages

- **DIO** (DODAG Information Option) — topology construction and maintenance
- **DAO** (Destination Advertisement Object) — route discovery, downward forwarding
- **DIS** (DODAG Information Solicitation) — external join request for a DIO

Modes of Operation

- **Storing** — each node stores routing table of its sub-DODAG
- **Non-storing** — only root has complete routing overview from DAOs



(a) Multipoint-to-point (MP2P)

(b) Point-to-point (P2P)

Figure 3: Traffic flows examples

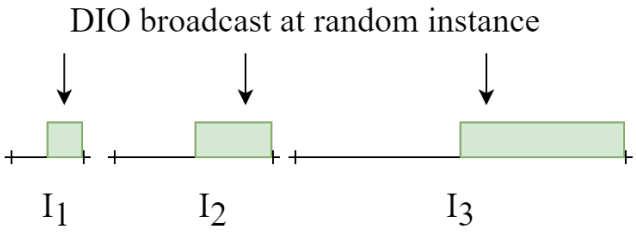


Figure 4: Trickle timer intervals

- Ensures fast topology convergence rate
- Minimizes messaging in stable network

RPL Implementation in INET

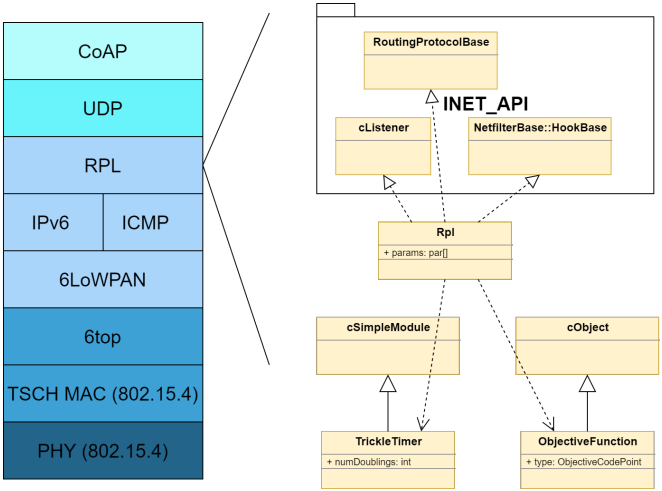


Figure 5: Rpl class diagram

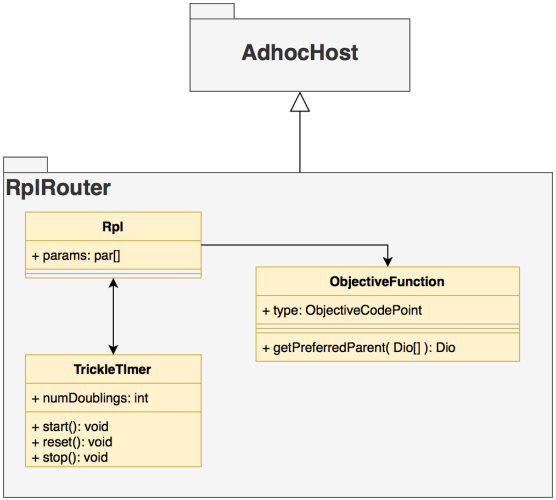


Figure 6: Rpl router module

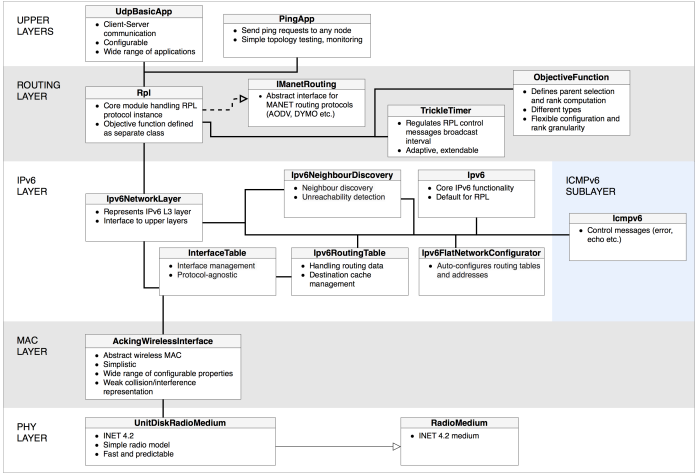


Figure 7: Rpl integration with INET

Table 2: RPL features implemented based on RFC 6550[3]

Control messages
DIO, DAO
Mode of operation
• Storing • Non-Storing
Traffic flows
• Multipoint-to-point (MP2P) • Point-to-multipoint (P2MP) • Point-to-point (P2P, storing only)
Extra
• Poisoning • Loop detection

Table 3: Rpl NED-configurable parameters

Name	Description
useBackupAsPreffered	Utilize backup parents to forward remaining packets
daoEnabled	Required for downward routing
storing	Storing\Non-Storing mode of operation
isRoot	Enable root capabilities
poisoning	Poison sub-DODAG upon detaching

1. DODAG construction
2. Multipoint-to-point traffic (static\dynamic)
3. Poisoning
4. Point-to-multipoint traffic
5. Point-to-point traffic
6. Loop detection
   - Rank error
   - Forwarding error

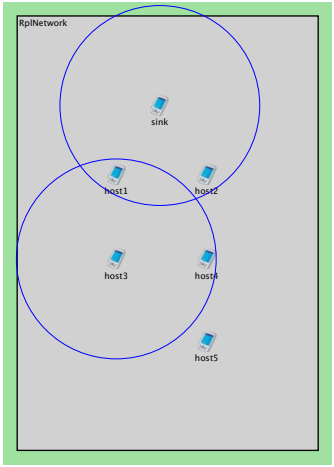


Figure 8: Simulation topology

Table 4: Simulation parameters

Parameter	Value
Communication range	150m
Distance between neighbors	≈120m
Transmission interval	uniform(1s, 10s)
Payload	56B
Mobility speed (dynamic scenario)	2mps

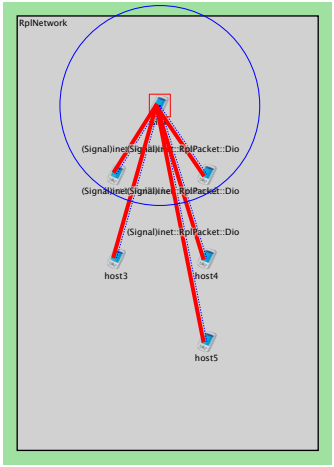


Figure 9: DODAG creation process

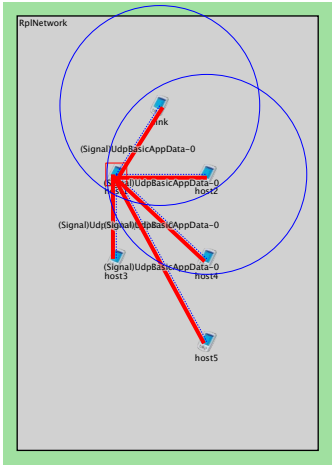


Figure 10: Multipoint-to-point traffic

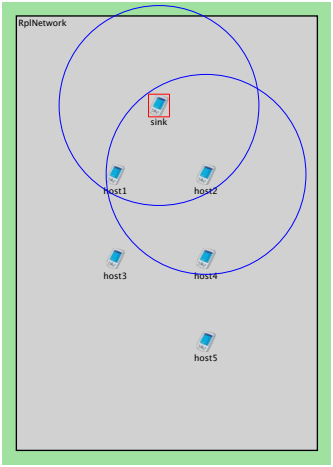


Figure 11: Multipoint-to-point traffic dynamic

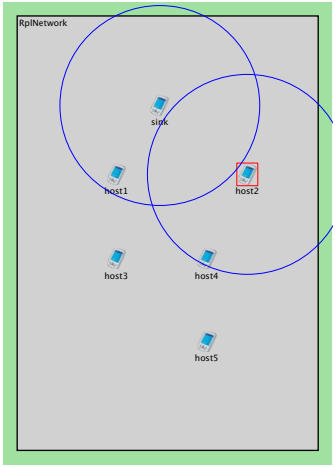


Figure 11: Multipoint-to-point traffic dynamic

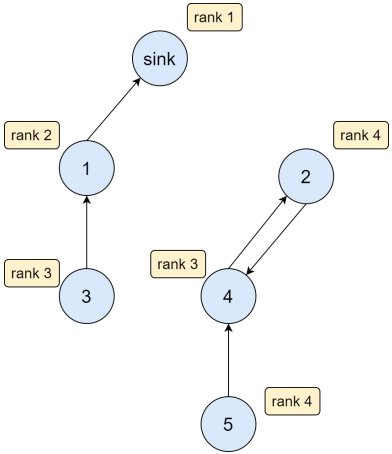


Figure 12: Dynamic scenario routing loop

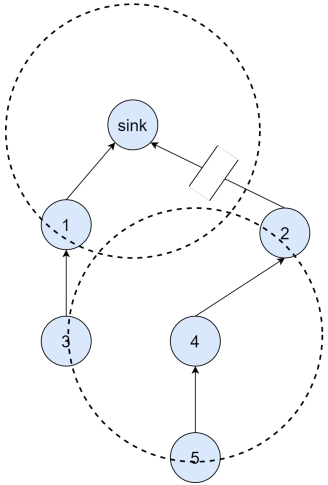


Figure 13: Sub-DODAG poisoning example

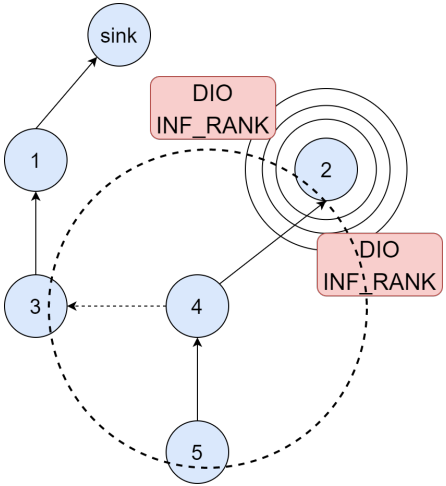


Figure 13: Sub-DODAG poisoning example

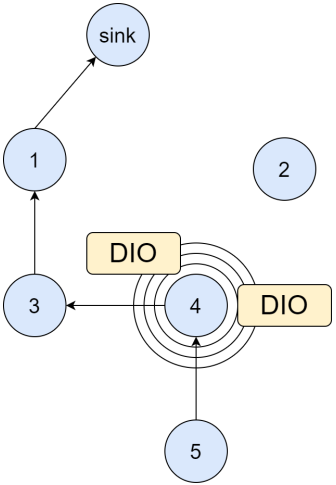


Figure 13: Sub-DODAG poisoning example

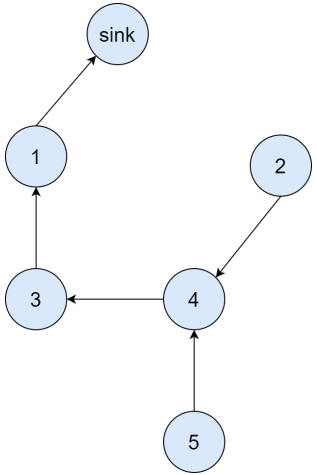
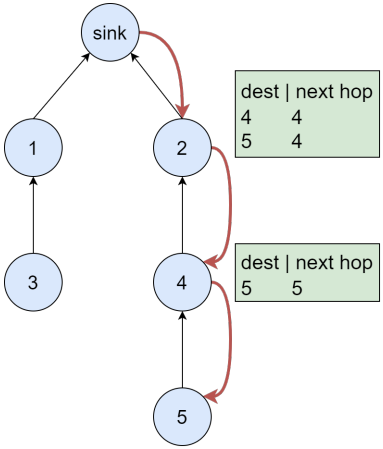
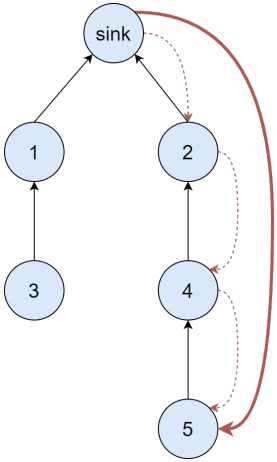


Figure 13: Sub-DODAG poisoning example



(a) Storing mode



(b) Non-Storing mode

Figure 14: Point-to-multipoint traffic flow example

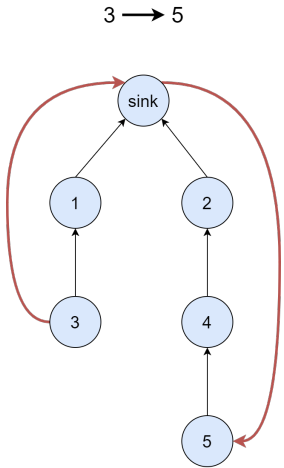


Figure 15: Point-to-point traffic flow example

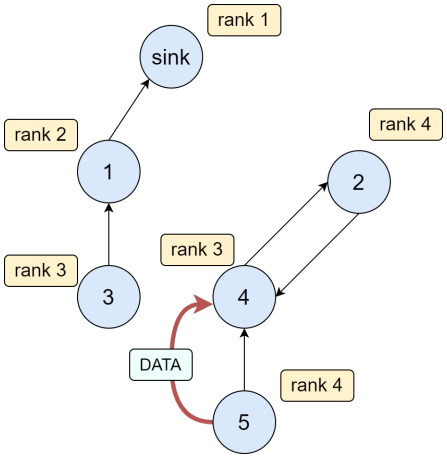


Figure 16: Rank error scenario

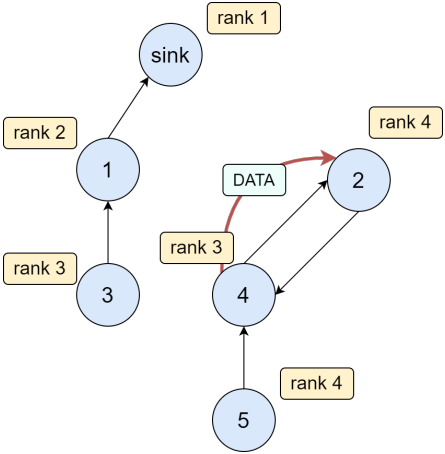


Figure 16: Rank error scenario

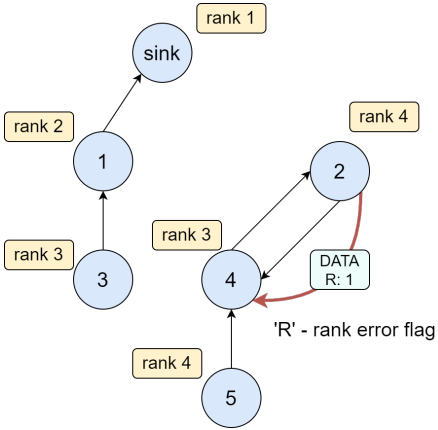


Figure 16: Rank error scenario

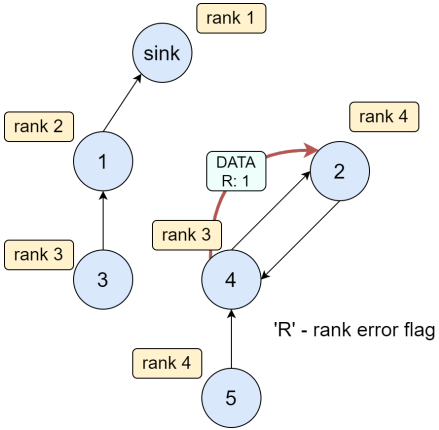


Figure 16: Rank error scenario

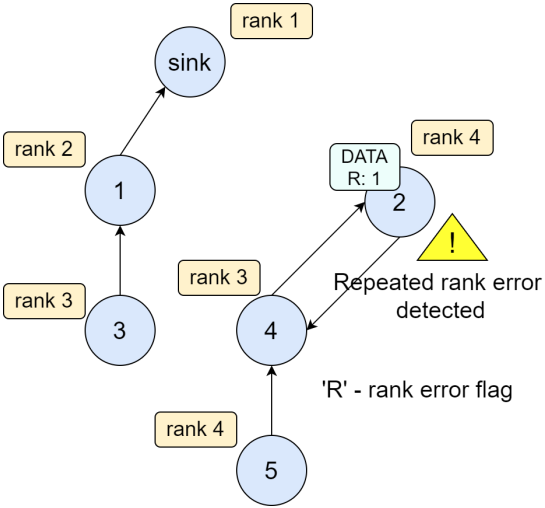


Figure 16: Rank error scenario

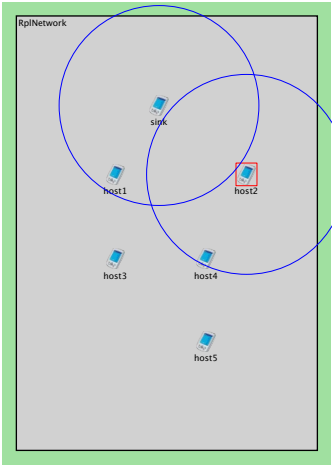


Figure 17: Forwarding error scenario

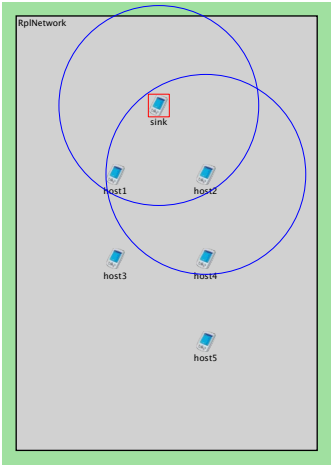


Figure 17: Forwarding error scenario

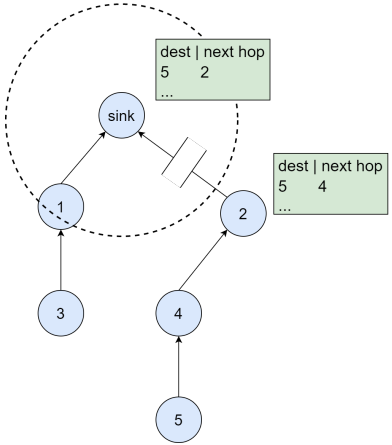


Figure 18: Forwarding error scenario

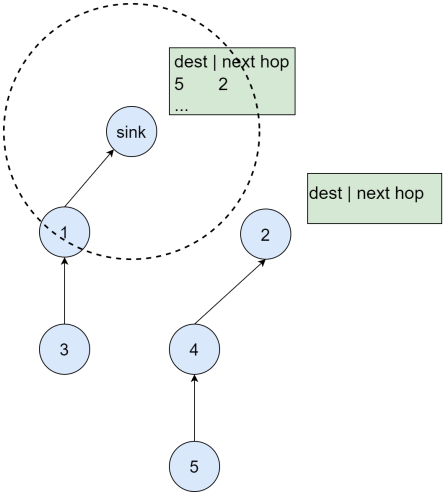


Figure 18: Forwarding error scenario

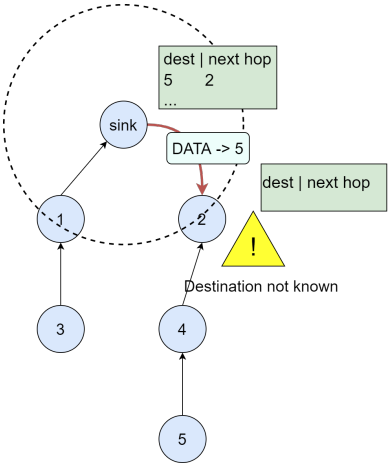


Figure 18: Forwarding error scenario

Conclusions

Conclusions

Achieved:

- RPL core functions according to RFC 6550
- Modular structure
- Basic showcase scenarios and unit testing

Goals for future work:

- Integrate with 6TiSCH stack
- Extend features and documentation per RFC 6550
- Set up CI & improve test coverage

[1] Oana Iova et al. "Rpl: The routing standard for the internet of things... or is it?"
In: *IEEE Communications Magazine* 54.12 (2016), pp. 16–22.

[2] Ming Zhao et al. "A comprehensive study of RPL and P2P-RPL routing protocols: Implementation, challenges and opportunities".
In: *Peer-to-Peer Networking and Applications* 10.5 (2017), pp. 1232–1256.

[3] Roger Alexander et al. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*. RFC 6550. Mar. 2012. DOI: 10.17487/RFC6550. URL: https://rfc-editor.org/rfc/rfc6550.txt.

Thank you for your attention!

Source code along with the shown scenarios available at:
https://github.com/ComNetsHH/omnetpp-rpl
(INET 4.2 and OMNeT++ 5.6.1/6.0-pre8 compatibile)