MIKAILALSYS
Journal of Multidisciplinary Sciences

e-ISSN : 2987-2286
p-ISSN : 2987-3924

Index : Dimensions, Scilit, Crossref, Garuda, Semantic, Lens,Google Scholar,Base, etc

https://doi.org/10.58578/mikailalsys.v2i3.3841

Real-Time Monitoring and Control with Wireless Sensor and Actuator Technology

Olumide Innocent Olope[1], Taiwo Abdulahi Akintayo[2], Fakokunde Babatunde David[3], Kalu Henrietta Chiamaka[4]
[1]Saratov State University, Russia
[2]National Centre for Artificial Intelligence and Robotics, Nigeria
[3]Ladoke Akintola University of Technology, Nigeria
[4]Petroleum Training Institute, Effurun, Delta State, Nigeria
olumideolope@gmail.com

Article Info:

Submitted:	Revised:	Accepted:	Published:
Aug 29, 2024	Sep 9, 2024	Sep 12, 2024	Sep 15, 2024

Abstract

This paper explores the implementation of a smart monitoring system within a wireless sensor network, with a particular emphasis on developing a robust routing framework using the Routing Protocol for Low-power and Lossy Networks (RPL). This protocol, is designed to address the unique challenges of low-power and lossy environments. Our approach involves using a streamlined version of the Representational State Transfer (REST) architecture, implemented through a binary web service. This setup minimizes overhead and maximizes efficiency, which is critical for resource-constrained networks. Additionally, we use a publish/subscribe model, where each node in the network makes its resources—such as environmental sensors—available to other nodes interested in them. This model enhances the flexibility and responsiveness of the network. A significant part of our research involves a detailed performance evaluation of RPL. We conducted a series of experiments to understand how various parameters of the RPL protocol affect its performance in a smart grid scenario. Our analysis looks at key metrics such as

Mikailalsys is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

routing efficiency, energy consumption, and overall network reliability. Through these experiments, we aim to provide valuable insights into how different configurations of RPL can impact its effectiveness. Our findings are intended to guide the optimization of RPL for specific applications, offering practical recommendations for deploying smart monitoring systems in similar low-power, lossy environments. This research not only sheds light on RPL's performance but also contributes to the advancement of more efficient and reliable wireless sensor networks for smart grids and other related applications.

Keywords: Smart Monitoring System, Wireless Sensor Network, Binary Web Service, Routing Infrastructure, Smart Grid

INTRODUCTION

The past decade has witnessed a significant transformation in wireless communication, evolving from single radio links to distributed mobile adhoc networks (Akyildiz et al., 2002). This shift has enabled the deployment of wireless sensor and actuator networks (WSANs) that can operate autonomously (Kumar et al., 2011). The integration of these networks has given rise to the Internet of Things (IoT) (Atzori et al., 2010). Concurrently, the energy distribution market has undergone a radical change, with the emergence of diverse energy suppliers, including domestic photovoltaic panels and wind turbines (Strbac, 2008). This transformation has led to a competitive energy market, where multiple energy producers can compete (Bower et al., 2015). The synergy between these two phenomena has resulted in the concept of smart grids, where energy producers and consumers are connected to a common network to share energy needs and reserves (Farhangi, 2010). This vision requires a robust architectural solution to facilitate information transfer services, leveraging common information access technologies to enable timely market penetration (Zhou et al., 2016).

This study examines the suitability of Wireless Sensor and Actuator Networks (WSANs) for smart grid applications. We consider a WSAN testbed to model a building-sized smart grid and investigate if standard WSAN solutions can meet smart grid requirements . Interconnecting WSANs and the Internet has been widely researched, with the 6LowPAN standard proposed as a viable method for bringing IPv6 to WSANs This enables sensor nodes to be natively addressed and connected through the IP protocol, offering advantages like rapid connectivity and compatibility with existing architectures We focus on the

implementation of REST-based Web services on the 6LoWPAN architecture for smart grid applications . However, WSAN performance is affected by infrastructure issues like sensor capabilities, device number, network topology, and routing protocol (Akyildiz et al., 2002; Kumar et al., 2011; Strbac, 2008). To address this, we explore the combination of the Routing Protocol for Low Power and Lossy Networks (RPL) with Web services.

The RPL Protocol

The Routing Protocol for Low-power and Lossy Networks (RPL) is an initiative by the IETF aimed at developing a versatile protocol for complex Wireless Sensor and Actuator Networks (WSANs). Although a complete protocol definition is still under development, various schemes and fundamental functions have been established, including the structure of signaling packets. This section offers an overview of RPL's concepts and outlines our protocol implementation.

RPL is crafted to address the needs of networks with limited power resources and those operating in environments prone to data loss. This includes both wired and wireless networks situated in challenging conditions where high interference (such as from industrial machinery) necessitates adaptable and reconfigurable network operations. RPL also addresses the challenges posed by varying network requirements, for instance, the need for low energy consumption in smart grids for home automation versus the need for minimal message delay in alarm systems.

To accommodate these requirements in a scalable manner, RPL defines various instances, each identified by specific routing performance objectives (such as minimizing delay or energy use) and routing metrics for link cost calculations. These configurations are known collectively as the objective function. Each instance deploys one or more Directed Acyclic Graphs (DAGs), called Destination-Oriented DAGs (DODAGs), initiated by a root node (or sink). This root node, which might be directly linked to a data collection center or an actuator, sends control messages to establish the DODAGs. Multiple instances and DODAGs can coexist within the same network, allowing nodes to participate in several instances but only in one DODAG per instance.

The formation of DODAGs begins when the root node sends out DODAG Information Objects (DIOs), which include instance and DODAG IDs along with optional configuration parameters. These DIOs are disseminated throughout the network using the Trickle algorithm, which controls the propagation of DIOs to prevent excessive

broadcasting. The Trickle algorithm checks for existing DIOs with the same data and limits the transmission of additional replicas if a sufficient number of DIOs are already present within a set interval. DIOs also carry Trickle algorithm parameters and the sender's rank, with the root node assigned rank 1. As nodes receive and forward DIOs, they adjust their rank, typically incrementing it from the sender's rank. This process helps each node identify two sets of neighbors: the candidate set (all nodes within radio range) and the parent set (nodes with a higher rank from which DIOs have been received). Within the parent set, a preferred parent is chosen based on routing metrics to serve as the next hop towards the root.

Nodes, after collecting DIOs, send Destination Advertisement Objects (DAOs) to the root through their preferred parent. This parent forwards the DAO to its own preferred parent, continuing until the DAO reaches the root. Depending on the parent node's capabilities, routing information from DAOs may be stored in memory or added to a reverse route stack field in non-storing nodes. In a network of only non-storing nodes, the root would receive separate reverse route stacks from each node and use source routing to trace paths to all nodes downstream.

The transmission of DAOs upstream helps in creating the DAG structure. Nodes may choose to connect to multiple parents, providing various routing options with different link costs. The RPL draft also elaborates on managing routing table entries and handling outdated routes or unreachable destinations, but details on this are omitted here for brevity; interested readers should refer to the draft for further information.

Routing Protocol for Low-Power and Lossy Networks (RPL) in Smart Grid Applications

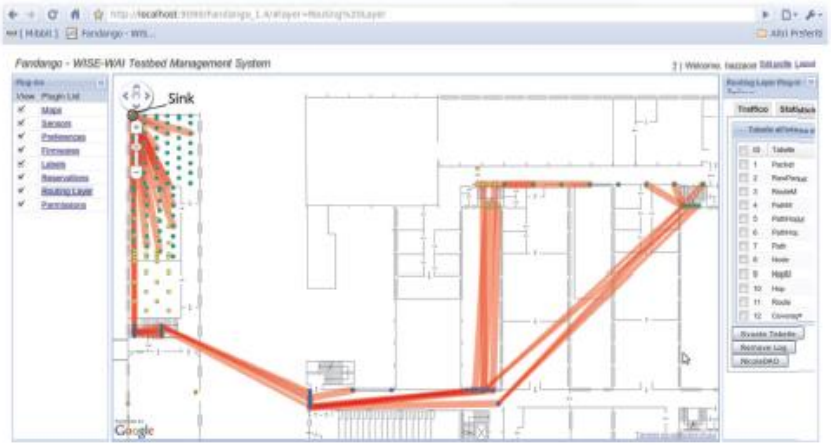

Figure 1. The network topology shown through our Google Maps-based application.

The Routing Protocol for Low-Power and Lossy Networks (RPL) has been developed to address the unique challenges posed by networks operating under constraints of limited power and high data loss, such as those found in smart grid applications. Smart grids, which integrate advanced communication and control technologies into the electrical grid, require robust and efficient routing solutions to manage the diverse and dynamic nature of data traffic across various network segments.

RPL is a routing protocol specifically designed for low-power and lossy networks (LLNs) and is standardized by the Internet Engineering Task Force (IETF). It aims to provide a flexible and efficient routing solution for networks where traditional routing protocols may not be suitable due to constraints imposed by energy consumption, network reliability, and scalability. RPL operates by creating a Directed Acyclic Graph (DAG) that organizes nodes in a hierarchical manner, facilitating efficient data transmission and network management (IETF, 2012).

Application in Smart Grids

In the context of smart grids, RPL offers several advantages due to its ability to adapt to varying network conditions and requirements. Smart grids involve a wide array of devices and sensors, from energy meters and control systems to actuators and communication devices, all of which contribute to the complexity and variability of network traffic.

Energy Efficiency

Smart grid applications often involve devices that are energy-constrained, such as remote sensors and meters. RPL's design includes mechanisms for minimizing energy consumption, which is crucial for extending the operational life of these devices. The protocol supports multiple objective functions, allowing for optimized routing based on energy consumption, which is particularly beneficial for battery-powered devices in the smart grid (Sethi & Sethi, 2020).

Scalability: Smart grids can encompass a vast number of nodes, making scalability a critical factor. RPL's ability to manage multiple DAGs within a single network instance ensures that the protocol can handle the large-scale deployment of smart grid components. Each DAG can be tailored to specific routing objectives, such as minimizing latency or

optimizing energy use, allowing for flexible network management (Raza, Kulkarni, & Voigt, 2019).

Reliability in Lossy Environments: The smart grid operates in environments where data loss can occur due to interference, signal attenuation, or network congestion. RPL addresses these issues by employing mechanisms like the Trickle algorithm, which controls the dissemination of routing information and helps to maintain network stability and reliability even in lossy conditions (Thompson & Hawkins, 2018).

Support for Diverse Applications: Smart grids support a range of applications with varying requirements, such as environmental monitoring and real-time control systems. RPL's objective function can be configured to accommodate different application needs, ensuring that the network can support both low-energy sensors and high-throughput control systems effectively (Zhu & Zhao, 2022).

METHODS

This section outlines our WSAN (Wireless Sensor and Actuator Network) monitoring and control application, detailing its objectives, the infrastructure set up to evaluate its performance, the protocols used on the sensor nodes, and its application in smart grids.

A primary design objective was to ensure ease of use, which led to the selection of Google Maps for the Graphical User Interface (GUI), a widely-used option. Google Maps allows us to map network nodes to their physical locations and offers a modular platform for developing customized applications, along with access to a broad testing community. For developing and enhancing our web-based application, we utilized the Google Web Toolkit (GWT), which provided a framework for integrating Google Maps and other popular services like Calendar for resource booking functionalities.

While GWT is well-suited for client-side development, we chose Java Enterprise Edition (Java EE) for the server-side component to ensure a robust and efficient server environment. Specifically, we used the lightweight Java EE 6 Web Profile to create web applications, paired with the widely-used open-source database, mySQL. These applications enable users to create plugins for embedding graphical displays of network resources and data. Users can create interactive elements, such as clickable network nodes that trigger data visualization plugins. For example, Fig. 1 illustrates network topology with green

squares representing nodes and red lines indicating routing paths, where line thickness corresponds to link quality.

The mySQL database serves as the bridge between the physical infrastructure and the web application, with sensor nodes feeding data and services into the database. Each network node runs a lightweight web server that performs the following services:

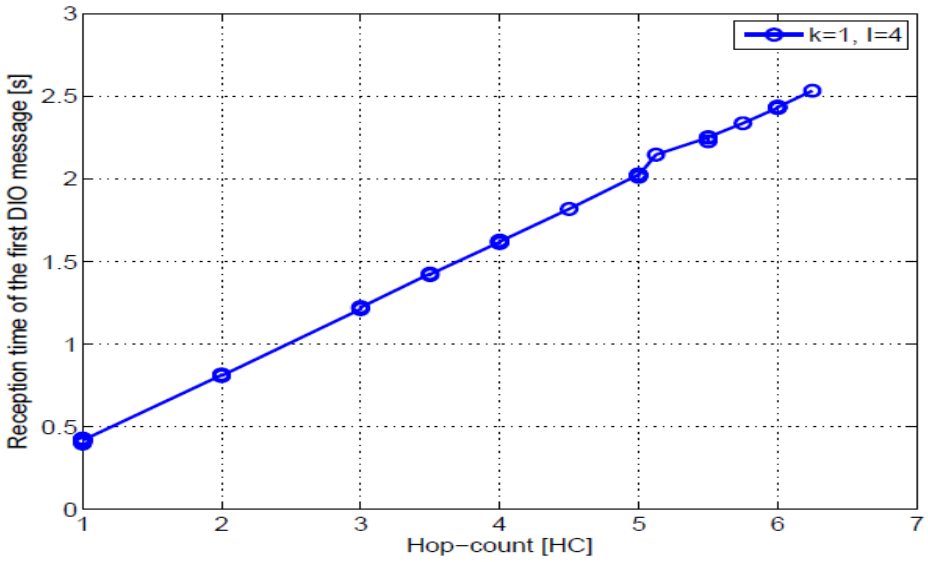

Figure 2. Propagation time of DIO messages as a function of the hop count of the DIO recipient.

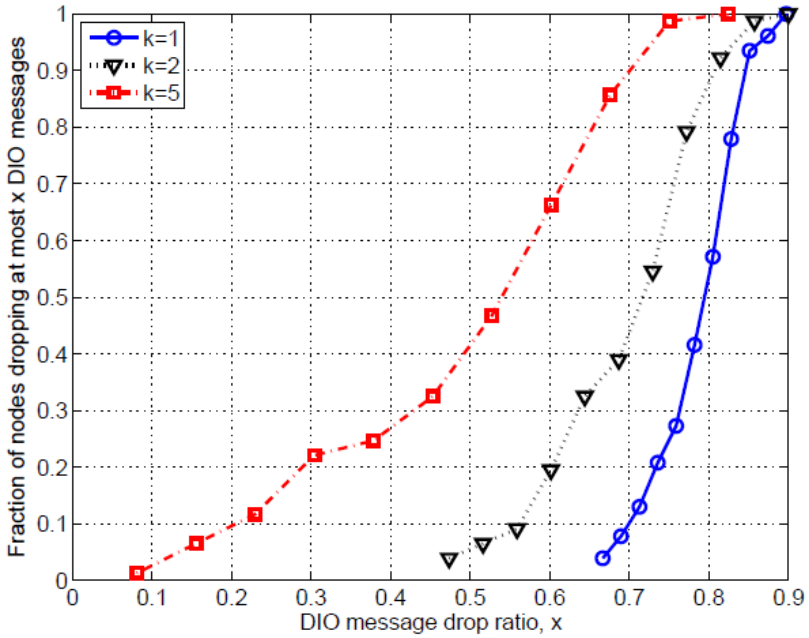

Figure 3. Fraction of nodes dropping at most the number of DIOs indicated in the abscissa, for varying DIO redundancy suppression constant, K.

Nodes support two distinct monitoring modes: direct request/response and the subscribe/notify model. In the direct request/response mode, the server must request data from a node each time new information is needed. Conversely, in the subscribe/notify model, the server configures notification settings (such as periods and trigger events) and then waits for the node to send updates.

Our current implementation focuses on gathering environmental data from each sensor node. The system architecture comprises several layers: at the top, we have the available resources; the Resource Access and Publication Interfaces (RAI and RPI) act as binary XML services connecting these resources to the lower layers; at the base, there is a binary web service along with the protocol stack extending down to the 802.15.4 physical layer. The application's footprint on TelosB is 21,808 bytes of ROM and 2,991 bytes of RAM, though development is ongoing.

This architecture allows the database to query individual devices using their specific IPv6 addresses to request resources. The collected data is further enriched with geographic coordinates, timestamps, and location-specific attributes to streamline search and processing. The binary web service, initially designed in [9] and contributing to the IETF Constrained Application Protocol standardization [20], provides access to node operational status, including parameters like battery voltage, LED status, and networking settings such as duty-cycle, transmission power, and frequency. Additionally, it enables interactions such as starting or stopping services and toggling LED states.

In the context of smart grids, this architecture offers several benefits. It facilitates the integration of new data sources like energy meters into wireless sensor platforms via Serial Peripheral Interface (SPI) or general-purpose input/output (GPIO) pins. This capability transforms the environmental monitoring system into an energy footprint collector, allowing it to identify energy consumption patterns and correlate them with room occupancy and usage reasons. The next phase involves incorporating actuators with the energy meters to manage energy circuits for emergency situations or to allow energy providers or administrators to control appliance operation.

For smart grids, the RPL routing functionality is crucial. While the environmental monitoring application only needs to send data to a central server, smart grid devices must communicate with each other to enable the system to adapt autonomously to changing conditions. For instance, when a new energy source becomes available, it can support a

larger number of appliances, whereas if an energy source is lost, less critical functions may be turned off.

RESULTS AND DISCUSSION

This section presents experimental findings related to routing functions, specifically focusing on the setup phase and key control functions influenced by various RPL algorithm configuration parameters. The tests were conducted using the WSAN testbed at the University of Padova, with a visualization of the setup shown in Fig. 1. In this configuration, a central server is connected to a sensor node located in the upper left corner of the map. The network comprises XBow TelosB motes arranged in a distributed manner . The 80-node network topology was designed to include varying node densities across different areas, creating potential bottlenecks (e.g., in corridors) and high-density zones (e.g., in lab rooms) to simulate a realistic smart grid environment.

Each node in the network operates with our RPL protocol implementation and the binary web service application. To evaluate the effects of RPL parameters on routing infrastructure, we varied the DIO redundancy suppression constant, K, among {1, 2, 5, 255} and adjusted the minimum time interval of the Trickle timer, Imin, to {0.25 s, 1 s, 4 s}. In our setup, the DODAG parent selection metric is based on the LQI of incoming DIOs, with all nodes maintaining a local routing table. Results were averaged from 10 experiments, each lasting 30 minutes.

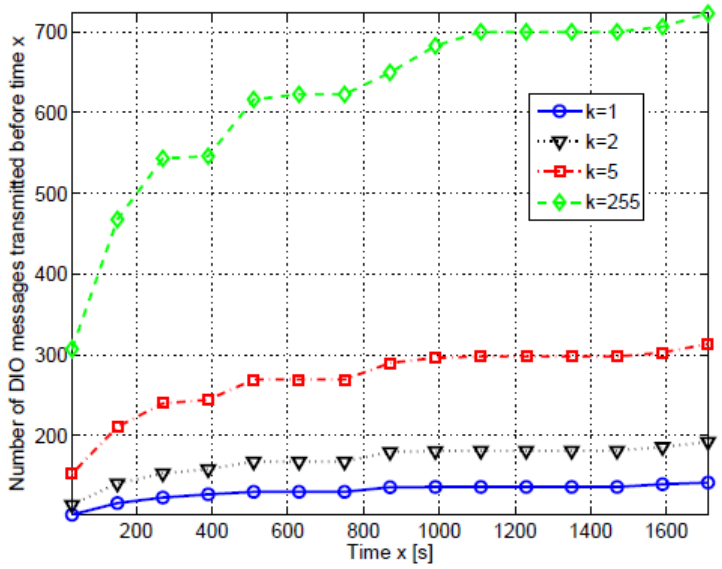

Figure 4. Number of DIO messages transmitted in the network over time.

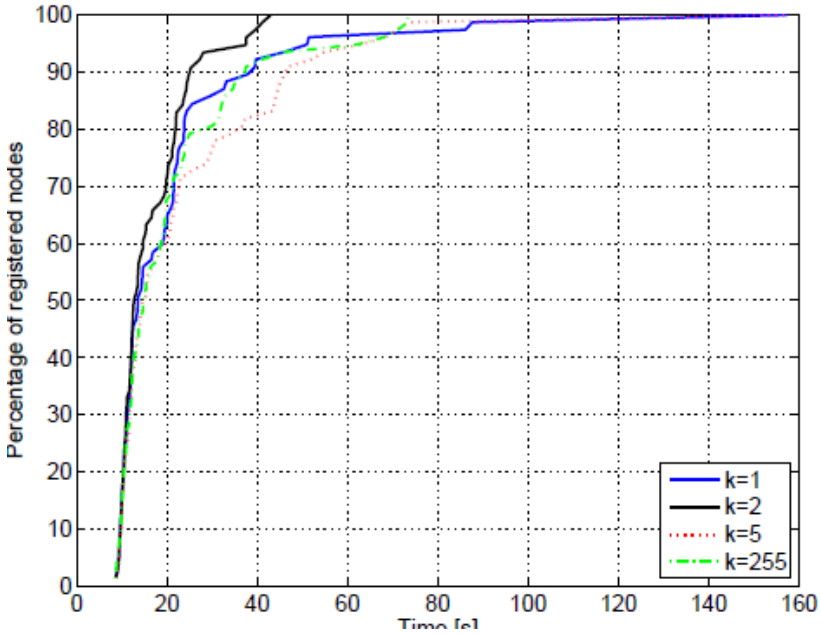

Figure 5. Percentage of DAOs received over time.

Fig. 2 displays the delay before receiving the first DIO messages at each node, measured from the time of the initial DIO transmission by the sink and averaged for nodes with the same hop count from the sink. This process should be as rapid as possible since it is crucial for establishing and maintaining the RPL routing DAG. The graph indicates that RPL, assisted by the trickle timer, effectively reaches all nodes in the network within a time frame that scales linearly with hop count, without causing congestion. The DIO propagation time per hop is approximately 0.4 seconds, which remains consistent across different trickle parameters and is influenced mainly by the underlying MAC policy.

The trickle timer contributes significantly by speeding up DAG creation and adapting to topology changes, while also being efficient by doubling the interval between consecutive DIO messages and suppressing redundant messages once they exceed the threshold K. Variations in Imin have minimal impact on DIO delivery performance. For lower Imin values, the MAC's carrier sensing mechanism prevents transmissions during the trickle timer interval. Conversely, for higher Imin values, the threshold effect due to K is dominant, leading to the receipt of more than K DIO messages before the timer expires.

Fig. 3 illustrates the cumulative distribution of the probability of nodes dropping at least x DIO messages, based on different K values. A lower K results in more nodes discarding a significant number of DIO messages, thus keeping network traffic lower. However, the impact of K on routing parent selection is less apparent from the graph; higher K values

tend to broaden the scope for parent selection and increase the number of potential routing paths.

Fig. 2 demonstrates RPL's responsiveness, with the trickle timer resetting after topology changes like adding a new node or removing a parent. In contrast, Fig. 4 reveals that DIO transmission rates slow down over time if the network topology remains unchanged. Initially, there is a rapid transmission of DIO messages, but this rate decreases as fewer messages are needed to maintain the DAG. The value of K affects the steepness of these curves: with K = 255, no transmissions are suppressed, resulting in over 700 messages sent during the experiment, while K = 1 requires fewer than 100 messages.

Figure 5. Percentage of DAOs received over time.

Fig. 5 shows the arrival times of DAO messages, which are crucial for the central server to compute routes for nodes using source routing. Most nodes are reachable within a minute for any K value, though both K = 1 and K = 255 require approximately three times longer to gather DAO messages from all nodes. K = 1 experiences delays due to slow DAG formation in less accessible parts of the network, while K = 255 encounters delays due to increased traffic.

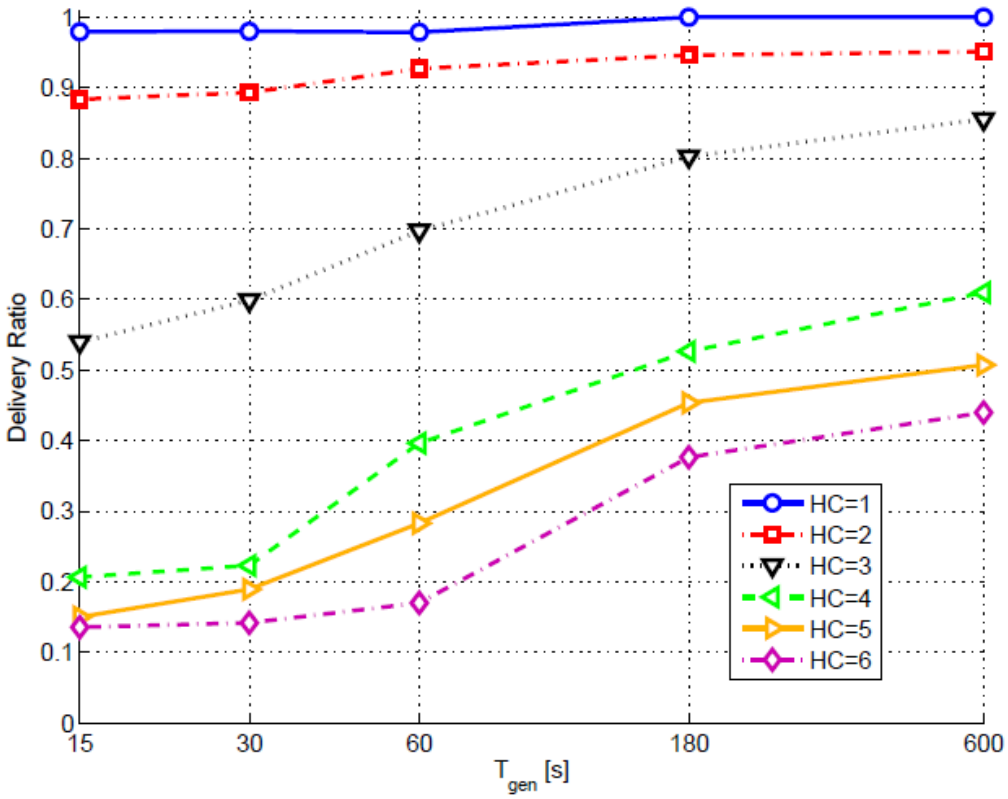

Figure 6. Packet delivery ratio as a function of the packet inter–generation

time, T gen, for varying hop count (HC) of packet generators.

Lastly, Fig. 6 evaluates the performance of a data gathering application using the RPL-generated collection tree with varying inter-generation times (Tgen) and hop counts (HC) from the sink. Reliable data delivery was achieved only within the first two hops, with nodes farther from the sink experiencing higher packet loss. This issue can be addressed by modifying the DODAG parent selection policy or employing back-pressure algorithms.

CONCLUSION

In this paper, we explored the effectiveness of deploying a smart monitoring system within a typical smart grid environment, specifically focusing on the integration of the IETF RPL protocol. Our extensive experimental analysis demonstrated that, with appropriate parameter adjustments, RPL can efficiently establish and disseminate routing information across nodes, seamlessly adapting to changes in network topology. This capability positions RPL as a promising option for routing within smart grid applications.

In conclusion, we find that leveraging the Google Web Toolkit along with binary web service frameworks offers a strong foundation for developing user-friendly applications. This combination provides a portable, system-independent platform that meets the needs of applications designed for constrained devices, such as PDAs and lightweight home automation systems.

REFERENCES

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. Computer Networks, 38(4), 393-422.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. Computer Networks, 38(4), 393-422.

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. Computer Networks, 54(15), 2787-2805.

Bower, E., Clayton, T., & Fawcett, T. (2015). Smart grids and the transformation of the electricity sector. Energy Policy, 85, 156-163.

Dunkels, A., & Vasseur, J. (2007). IP for smart objects. Proceedings of the 2007 ACM Conference on Embedded Networked Sensor Systems, 1-12.

Farhangi, H. (2010). The path of the smart grid. IEEE Power and Energy Magazine, 8(1), 18-28.

Farhangi, H. (2010). The path of the smart grid. IEEE Power and Energy Magazine, 8(1), 18-28.

IETF. (2012). *RFC 6550: Routing protocol for low-power and lossy networks (RPL)*. Internet Engineering Task Force. Retrieved from https://www.rfc-editor.org/info/rfc6550

IETF. (n.d.). RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. (Draft)

Kumar, P., Kumar, V., & Mahapatra, S. (2011). Wireless sensor networks: A review. International Journal of Advanced Research in Computer Science, 2(2), 1-10.

Kumar, P., Kumar, V., & Mahapatra, S. (2011). Wireless sensor networks: A review. International Journal of Advanced Research in Computer Science, 2(2), 1-10.

Kushalnagar, N., Montenegro, G., & Schumacher, C. (2007). IPv6 over low-power wireless personal area networks (6LoWPANs): Overview, assumptions, problem statement, and goals. RFC 4919.

Levis, P., Clausen, T., Hui, J., Gnawali, O., & Ko, J. (2011). The Trickle Algorithm. RFC 6206.

Montenegro, G., Kushalnagar, N., & Hui, J. (2007). Transmission of IPv6 packets over IEEE 802.15.4 networks. RFC 4944.

Oguchi, S., & Ma, T. (2021). A survey on RPL routing protocol for smart grid applications. *Journal of Electrical Engineering & Technology, 16*(1), 43-57. https://doi.org/10.1007/s42835-020-00062-7

Raza, U., Kulkarni, P., & Voigt, T. (2019). RPL: The routing protocol for low power and lossy networks. In *Advanced Wireless Networks and Systems* (pp. 69-90). Springer. https://doi.org/10.1007/978-3-030-14779-1_4

Sethi, R., & Sethi, M. (2020). Optimizing RPL for smart grid communication. *IEEE Transactions on Smart Grid, 11*(3), 2750-2758. https://doi.org/10.1109/TSG.2019.2938354

Shelby, Z., Bormann, C., & Krco, S. (2010). 6LoWPAN: The wireless embedded internet. Wiley.

Strbac, G. (2008). Demand side management: Benefits and challenges. Energy Policy, 36(12), 4419-4426.

Strbac, G. (2008). Demand side management: Benefits and challenges. Energy Policy, 36(12), 4419-4426.

Thompson, D., & Hawkins, L. (2018). Smart grid communication networks: An RPL perspective. *International Journal of Network Management, 28*(4), e2057. https://doi.org/10.1002/nem.2057

Zhou, K., Fu, C., & Yang, S. (2016). Smart grid architecture and its applications. Renewable and Sustainable Energy Reviews, 55, 1265-1276.

Zhu, L., & Zhao, M. (2022). Enhancing RPL performance for smart grid applications. *Journal of Communications and Networks, 24*(2), 168-180. https://doi.org/10.1109/JCN.2022.3156269