Enhancing LOADng Routing Protocol for G3-PLC Networks

Jiazi Yi

École Polytechnique
Route de Saclay, Palaiseau, 91128, France
e-mail: jiazi.yi@polytechnique.edu

Abstract

Lightweight On-demand Ad Hoc Distance-vector Routing – Next Generation (LOADng) is ITU-T G.9903's default link layer routing protocol for G3-PLC and shall be supported by ITU-T G.9903 devices. It is a light-weight routing protocol with simple operations that are suitable for low power and lossy networks. On the other hand, designed with a modular approach, it can be extended to improve the performance in specific network scenarios. This paper introduces different extensions proposed for LOADng, including the *Smart Route Request*, *Expanding Ring Search*, *Collection Tree* and *Depth First Forwarding*. Those extensions mainly aim at reducing the route discovery cost and improving the packet delivery ratio in constrained environments. Their tradeoffs are also discussed.

Index Terms

LOADng, G3-PLC, routing, protocol extensions

I. Introduction

"Lightweight On-demand Ad hoc Distance-vector Routing Protocol – Next Generation (LOADng)" [1] is a derivative of Ad hoc On-demand Distance Vector protocol (AODV) [2] to enable efficient, scalable and secure routing in constrained environments. It is accepted as part of an update to the G3-PLC ITU-T standard for communication in the "smart grid" [3].

Designed for constrained environments, LOADng has a lightweight core specification that work in common network environments and data traffic patterns [4]. On the other hand, based on the deployments and experiences of LOADng, different extensions are proposed to enhance the performance of LOADng routing protocol in specific networks.

This paper introduces and summarizes the extensions proposed for LOADng, which include:

- *Smart Route Request* that aims at improving the route discovery efficiency based on existed routing information;
- *Expanding Ring Search* that aims at limiting the need of network-wide dissemination of route request messages;
- *Collection Tree Protocol* that aims at reducing the routing overhead if a collection tree is desired;
- *Depth First Forwarding* that aims at improving the data delivery reliability over lossy links.

The remainder of this paper is organized as follows: section II gives an overview of LOADng core specification. Section III, V, IV and VI introduce different extensions respectively. For each extension, the basic mechanism, expected performance enhancement, applicability and the costs of such extension (*i.e.,* tradeoffs) are discussed. Section VII concludes this paper .

II. LOADng Overview

LOADng inherits the basic protocol operations from AODV: on-demand generation of Route Requests (RREQs) by a router (originator) for discovering a path to a destination, forwarding of such RREQs until they reach the destination router, generation of Route Replies (RREPs) upon receipt of an RREQ by the indicated destination, and unicast hop-by-hop forwarding of these RREPs towards the originator. If a path is detected broken, *i.e.,* if forwarding of a data packet to the recorded next hop on the path to the destination is detected to fail, local path repair can be attempted, or a Route Error (RERR) message can be returned to the originator of that data packet.

LOADng has been designed with the philosophy of a *minimal core*, containing a small set of protocol operations and implementation requirements lending itself to a simple implementation with a small code footprint, as well as small operational state requirements. This minimal core is, at the same time, carefully crafted so as to enable extensions (when needed) to be developed, and deployed, in a fashion remaining interoperable with this minimal core. Compared to AODV, LOADng has following characteristics:

- **Modular design:** The core specification defines the simple and lightweight core functions of the protocol. LOADng is extensible, by way of a flexible packet format permitting addition of arbitrary attributes and information via new message types and/or TLV (Type-Length-Value) blocks.
- **Optimised Flooding:** Reducing the overhead incurred by RREQ forwarding. Jitter is employed, to reduce the probability of losses due to collisions on lower layers [5].

- **Flexible Addressing:** Address lengths from 1-16 octets are supported[1]. The only requirement is that within a given routing domain, all addresses are of the same address length.
- **Metrics:** Different metrics are supported, to make use of link information from different layers.
- **Destination-Replies:** Intermediate LOADng Routers are explicitly prohibited from responding to RREQs, even if they may have active routes to the sought destination.
- **Reduced state:** A LOADng Router is not required to maintain a precursor list, thus when forwarding of a data packet to the recorded next hop on the path to the destination fails, an RERR is sent only to the originator of that data packet.

III. Smart Route Request

Reducing the overhead, delay and complexity of the Route Discovery process (RREQ/RREP exchange) is key to adapting on-demand routing protocols for use in constrained environments. AODV includes an "intermediate/gratuitous RREP mechanism", which attempts at reducing this overhead and complexity – a mechanism which has been eliminated from LOADng. This section discusses the rationale for LOADng eliminating "intermediate/gratuitous RREPs", and presents an alternative mechanism denoted *Smart Route Requests* (SmartRREQ). The SmartRREQ mechanism attains a performance comparable to that of "intermediate/gratuitous RREPs" from AODV, while incurring smaller protocol messages, simpler protocol message processing, and offers advantages with respect to securing routing protocol operations.

During the Route Discovery process of AODV, an intermediate router can generate an intermediate RREP in response to an RREQ if it has a valid path to the destination sought – and must, if so, also generate a gratuitous RREP and send this to the desired destination in order to establish a complete and bi-directional route. In order to avoid routing loops when permitting intermediate routers to generate intermediate RREPs, an RREQ in AODV carries an RREQ ID, destination sequence number, and originator sequence number in RREQ messages – recorded and maintained by intermediate routers, and used for when processing RREQs and RREPs.

In LOADng, even if a LOADng Router already has an available and valid paths to the destination, intermediate RREPs are prohibited, so as to reduce the control message size and, still, guarantee loop freedom. For a LOADng Router, receiving an RREQ:

- if it is the ultimate destination, it must respond by an RREP message;
- or else, if it is an intermediate LOADng Router, it has to rebroadcast the RREQ, even if it has a valid path to the destination.

[4] shows that this simplification of LOADng renders the protocol more adapted to constrained environments, attaining lower routing overhead and fewer collisions.

In some network types, such as sensor networks, it is common to have sensor-to-root (multipoint-to-point – or MP2P) traffic. While eliminating intermediate RREP can reduce the size of control message and simplify the protocol process, the side effect of blind flooding RREQ cannot be ignored in this kind of scenarios.

SmartRREQ is thus proposed to replace intermediate route reply while retaining the loop-freedom nature and security mechanism of LOADng. When SmartRREQ is used, a LOADng Router initiates a Route Discovery by broadcasting an RREQ message with *smart-rreq* flag set (denoted *RREQ_SMART* message).

On receiving an *RREQ_SMART* message, an intermediate (*i.e.,*, which is neither the source nor destination.) LOADng Router performs the following procedures:

1) If the intermediate LOADng Router has a valid path to the destination, AND the next hop field of the corresponding routing tuple is not equal to the previous hop address of the RREQ, then the *RREQ_ SMART* is unicast to the next hop.
2) Otherwise the *RREQ_SMART* is broadcast, as usual, to all its neighbors.

Simulation results illustrated in [6] show that this extension can considerably reduce the routing overhead in common scenarios (*e.g.,* P2P traffic), and is especially efficient if most of the LOADng Routers are sending data packets to a few common destinations in the network (*e.g.,* MP2P traffic). This reduced overhead is obtained without punishing other performance metrics, such as data delivery ratio, average end-to-end delay, etc. – thus, rendering this extension highly recommended especially in data collection scenarios.

IV. Expanding Ring Search

The Expanding Ring flooding for path discovery was firstly used in AODV as an experimental option, in an attempt to limit the need for network-wide dissemination of RREQs. Then it is extended to LOADng.

A router will at first send an RREQ with a reduced TTL (Time-To-Live) – causing the RREQ to not be flooded through the entire network, but only up to a limited distance. If the destination sought receives the RREQ, or an intermediate router has a path to the sought destination, an RREP (possibly intermediate/gratuitous) is generated and a network-wide flooding is avoided. If no RREP is received by the originator in expected delay, another RREQ message is, after a brief delay, generated with increased TTL to eventually cover the entire network. More details of the extension specification can be found in [7].

[1] *i.e.,* IPv6, IPv4, 6LowPAN short addresses, Layer-2 MAC addresses etc. are all supported by LOADng

Note that while this may be an advantage in some cases, this mechanism can also be a double-edged sword, and cause increased rather than decreased control traffic: if no router closer to the originator of an RREQ than the final destination has a path to the destination, more control traffic is generated by such repeated Expanding Ring floods due to multiple generation of RREQ messages.

The simulation results [7] show that the reduction of overhead comes at the expense of increasing Route Discovery delay, especially in point-to-point scenarios, where there is no "single" destination in the network.

Therefore, this extension is applicable only if there are a few "common" destinations in the network, or where delays are non-crucial. It is not recommended in scenarios where destinations are sparsely and randomly distributed in the network, such as remote control, or scenarios in which response time is a critical metric, such as alarms.

V. Collection Tree Protocol

The LOADng core specification, and its extensions like SmartRREQ and Expanding Ring, introduced in previous sections, find paths between any originator-destination pairs. This kind of point-to-point traffic pattern matches the basic traffic model of the Internet. However, in many deployments of LLNs, another important traffic pattern, called sensor-to-root, or multipoint-to-point, exists. In such traffic scenarios, there is one or more devices that plays the role as "root" – data sink for all traffic – and where and all the other devices in the network communicate with the root. If paths from all the other devices to the root are required, it is more efficient to build a "collection tree", which is a directed graph that all edges are oriented toward and terminate at one root router, and it is to discover and maintain the set of point-to-point routes from them all to that "root".

The collection tree extension aims at building bi-directional routes between the root router and all other routers. The collection tree is built by way of the following procedures – initiated by the LOADng Router wishing to be the root of the collection tree:

a) Collection Tree Triggering (by the root router): The root router generates a *collection tree trigger* RREQ message (denoted RREQ_Trigger). Both the originator and destination of the *RREQ_Trigger* are set to an interface address of the root. The RREQ_Trigger is disseminated as normal RREQ messages. When a RREQ_Trigger is generated, a *collection tree build* RREQ message (denoted RREQ_BUILD) is also scheduled.

b) Bi-directional Neighbor Discovery: This step is to blacklist the uni-directional neighbors. On first receiving of an RREQ_Trigger, a HELLO message is triggered carrying the neighbor information that a router just learnt. Based on the RREQ_Trigger and the HELLO message exchange, each router is able to identify its bi-directional neighbors.

c) Collection Tree Building: After the bi-directional neighbor discovery, the root router generates an RREQ_BUILD and disseminate the message in the network. The routers only processes the RREQ_BUILD messages received from bi-directional neighbors, and set up the routing table entry to the root accordingly.

Thus, each LOADng Router will record a path to the root, and this path will contain only bi-directional links; the collection tree is built, enabling upward traffic.

If paths from the root to other routers (sensors) inside the network is required, each LOADng Router receiving an RREQ_Build will unicast an RREP to the root, transmitted and processed according as normal RREP message. Thus, downward traffic is also enabled.

When the collection tree is being built, control messages may get lost. Thus, some LOADng Routers may not be included in the initial collection tree because of transient transmission failure of collection tree building messages. Furthermore, the routing entries may expire because of not being updated in a timely fashion. Both of those result in that the path to the root is not available in some of the LOADng Routers.

In this case, a LOADng Router with data traffic to send to the root will initiate Route Discovery, according to the usual procedures LOADng core specification. To avoid that RREQ being broadcast through the whole network, and to benefit from the fact that "most of other neighbor routers might have an available route to the root", the SmartRREQ extension introduced in section III or/and the Expanding Ring extension introduced in section IV can be employed.

When a link on an active path to a destination is detected as broken (by way of inability to forward a data packet towards that destination), an RERR (route error) message is unicast to the source of the undeliverable data packet an may trigger a new Route Discovery.

The collection tree extension for LOADng is designed to build multipoint-to-point paths with reduced overhead. For a network with n routers, the message overhead is $O(n)$ for building a connection tree compared to $O(n^2)$ in LOADng, as shown in simulation results in [8] [9]. Although the collection tree extension introduces new message flags and types, it is interoperable with the LOADng core protocol: a LOADng Router without collection tree extension can join the collection tree by initiating a Route Discoveryto the root, and can participate as an internal node in the collection tree by way of the same mechanism. Further details of the specification can be found in [8] [9].

VI. Depth First Forwarding

Routing protocols for LLNs, such as LOADng, are typically designed to limit the routing overhead imposed to networks as much as possible, and to be adapted to the varying nature of communication media. However, even once paths have been found, these paths may be unusable from time to time due to different reasons: presence of noise or interferences, low power supply in certain devices, uni-directional links, etc. From a routing protocol point of view, when such link failure is detected, it needs some extra signaling and/or time to recover and discover new, valid paths. During this recovery phase, data packets being sent over the broken link must either be buffered and wait for the path recovery, or be dropped because of lack of memory in constrained devices.

"Depth-First Forwarding in Unreliable Networks" (DFF) [10] is an experimental data forwarding standard by the IETF, which proposes a mechanism for rapid and localized recovery in case of link failure. Colloquially speaking, if a device fails in its attempt to forward a packet to its intended next-hop, then DFF suggests a heuristics for "trying another of that devices' neighbors", while keeping track of (and preventing) packet loops.

LOADng is extended to support DFF by adding neighbor discovery mechanisms using HELLO messages. Each router can thus maintain a list of all its bi-directional neighbors. When a packet is to be forwarded by a router using DFF, the router creates an ordered list of *Candidate Next Hops* for that packet. DFF proceeds to forward the packet to the first next hop in the list. If the transmission was not successful (as determined by the underlying link layer) or if the packet was "returned" by a next hop to which it had been sent before, the router will try to forward the packet to the subsequent next hop on the list based on "depth-first searching". A router "returns" a packet to the router from which it was originally received once it has unsuccessfully tried to forward the packet to all elements in the "Candidate Next Hop List" (CNHL). If the packet is eventually returned to the originator of the packet, and after the originator has exhausted all of its next hops for the packet, the packet is dropped. A detailed specification can be found in [11] [12].

The simulation results show that the DFF extension for LOADng can effectively increase the data delivery reliability in lossy networks [11] [12]. It can obtain a 20 percentage point gain on data delivery ratio. That said, the side effects of applying this extension are also obvious: the depth-first search yields much longer data delivery delays and paths. The DFF extension is therefore recommended for scenarios with lossy channels, and where data traffic is (at least, somewhat) delay tolerant. However, this extension should not be used if the channel is not particularly lossy – in which case DFF will not be producing any benefits, but will consume network and other resources for bi-directional neighbor discovery.

VII. Conclusion

This paper introduces four extensions of LOADng routing protocol: *Smart Route Request*, *Expanding Ring Search*, *Collection Tree Protocol* and *Depth First Forwarding*.

The Smart Route Request and Expanding Ring Search can make use of the existed routing information to reduce network-wide RREQ broadcast. The Smart Route Request is with little cost, it is thus recommended in most of the scenarios. The Expanding Ring Search is beneficial when the traffic in the network is destined to some common destinations, and might increase the route discovery delay. The Collection Tree Protocol is to build a collection tree using bi-directional links. It can effectively reduce the route discovery overhead for the multipoint-to-point and point-to-multipoint traffic patterns. Depth First Forwarding works on the forwarding plane, which tries to "save" the packet in case of link failure before the routing table get recovered. DFF can effectively increase the packet delivery ratio at the cost of increasing the neighbor discovery overhead, path length and end-to-end delay.

References

[1] T. Clausen, A. C. de Verdiere, J. Yi, A. Niktash, Y. Igarashi, H. Satoh, U. Herberg, C. Lavenu, T. Lys, and J. Dean, "The Lightweight On-demand Ad hoc Distance-vector Routing Protocol – Next Generation (LOADng)," Internet Draft, work in progress, draft-clausen-lln-loadng, IETF, July 2013.

[2] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," Experimental RFC 3561, July 2003.

[3] ITU, "ITU-T G.9903: Narrow-band orthogonal frequency division multiplexing power line communication transceivers for G3-PLC networks: Amendment 1," May 2013.

[4] T. Clausen, J. Yi, and A. C. de Verdiere, "LOADng: Towards AODV Version 2," in *VTC Fall*. IEEE, 2012, pp. 1–5.

[5] J. A. Cordero, J. Yi, and T. Clausen, "An adaptive jitter mechanism for reactive route discovery in sensor networks," *Sensors*, vol. 14, no. 8, pp. 14 440–14 471, 2014.

[6] J. Yi, T. Clausen, and A. Bas, "Smart Route Request for On-demand Route Discovery in Constrained Environments." Proceedings of the IEEE International Conference on Wireless Information Technology and Systems, September 2012.

[7] A. Bas, J. Yi, and T. Clausen, "Expanding Ring Search for Route Discovery in LOADng Routing Protocol." Proceedings of The 1st International Workshop on Smart Technologies for Energy, Information and Communication, September 2012.

[8] J. Yi, T. Clausen, and A. C. de Verdiere, "Efficient Data Acquisition in Sensor Networks:Introducing (the) LOADng Collection Tree Protocol." Proceedings of the 8th IEEE International Conference on Wireless Communications, Networking and Mobile Computing., October 2011.

[9] J. Yi and T. Clausen, "Collection tree extension of reactive routing protocol for low-power and lossy networks," *International Journal of Distributed Sensor Networks*, vol. 2014, 2014.

[10] U. Herberg, A. Cardenas, T. Iwao, M. Dow, and S. Cespedes, "Depth-First Forwarding (DFF) in Unreliable Networks," RFC 6971, IETF, June 2013.

[11] J. Yi, T. Clausen, and U. Herberg, "Depth first forwarding for low power and lossy networks: Application and extension," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 462–467.

[12] Yi, T. Clausen, and U. Herberg, "Depth-first forwarding for unreliable networks: Extensions and applications," *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 199–209, 2015.