

# Babel Multi-hop Routing for TinyOS Low-power Devices

Antoine Hauck  
*Distributed Secure Software Systems*  
*Lucerne University of Applied Sciences and Arts*  
*Horw, Switzerland*  
*Email: antoine.hauck@hslu.ch*

Peter Sollberger  
*CC Electronics*  
*Lucerne University of Applied Sciences and Arts*  
*Horw, Switzerland*  
*Email: peter.sollberger@hslu.ch*

**Abstract**—Efficient routing in Wireless Mesh networks (WMN) with limited bandwidth is a challenging task, especially in networks where nodes have restricted resources. In such environments routing mechanisms should have a small footprint, low CPU usage and minimal routing overhead. If nodes are mobile, topology changes occur permanently, so the routing protocol has to converge fast and remain loop-free. Traditional routing protocols for IP-based wired networks have in many aspects been proven inadequate for WMNs. Therefore protocols, like Destination-Sequenced Distance Vector (DSDV) and Ad-hoc On-demand Distance Vector (AODV) routing, has evolved to overcome the difficulties of WMNs. One of the most recent is Babel, a proactive distance-vector protocol with reactive features based on DSDV and AODV. Due to its specific characteristics, Babel should be able to run efficiently on WMNs and low-power devices. A stable Babel routing daemon exists for Linux and Mac OS X. This Work in Progress paper outlines a simplified subset implementation of the Babel routing protocol without using IP, especially designed to fit into low-power and hardware constrained wireless devices which are running TinyOS.

**Keywords**-Wireless mesh networks; Routing protocols; Low power electronics; Embedded software;

## I. INTRODUCTION

In WMNs, usually, nodes can not exchange data directly with certain other nodes, because they are not within the sender's range. With the aid of routing protocols the range of communication can be extended when intermediate nodes are used to forward a packet within a well known-path to the destination. Due to the mobility in networks, links between nodes are continuously being established and broken. Routing protocol mechanisms must be able to react in case of a broken link or failed node and propagate the topology change in the network efficiently.

While writing this paper, our institute is working on the European Union FP7 HydroNet [22] project. The HydroNet project aims at designing, developing and testing a new technological platform for improving the monitoring of water quality. It is based on a network of floating buoys and unmanned catamarans. Both are equipped with sensors to measure the pollution. Wireless communication in the HydroNet network is used to obtain and provide steering, position and sensor data between various nodes. Around twelve nodes are intended to monitor a sea area of 10 km x

3 km. In many cases, a direct point-to-point communication between two nodes is not possible, either the distance between them might exceed the maximum radio range or obstacles disturb or reflect the radio signals. In HydroNet every node is not only a data source or sink but also has to act as a router. A communication infrastructure based on the IEEE 802.11 standards was inadequate due to power constraints, distance limitations and regulations. Therefore, a Tinynode [11] like low-power device was developed with a TI MSP430 8 MHz micro controller, a Semtech XE1205 radio running at 434 MHz and a 2.5 W (34 dBm) amplifier to overcome greater distances. The operating system running on this node is TinyOS, an open source operating system designed for low-power wireless sensor networks. The node is connected via RS232 to the robot main controller. The nodes are not interconnected to other networks and also do not require global IP connectivity. Flat routing, as described in section III-B, will suffice.

Traditional routing protocols such as RIP [12] or OSPF [13] were designed for wired networks and update too infrequently to deal with the constant mobility in WMNs [14]. It must be considered that WMNs use a shared medium, usually with low bandwidth and unreliable transport characteristics, so routing loops can occur by lost updates due to collisions [20], noise or flooded links. Additionally low-power devices have very restricted resources and their lifetime is often limited to the battery capacity and therefore a routing protocol must be economic and simple. Routing table entries and any data that needs to be exchanged and stored on the device, to maintain the routing operation, must be kept to a minimum to fit into the limited capacity. Murray, Dixon and Koziniec proved that the routing protocol's overhead is the largest determinant of performance in WMNs and that the OSI layering has little impact [3]. It is substantial for ad hoc routing protocols to provide mechanisms to reduce these overheads. As an experimental comparison of three multi-hop ad hoc routing protocols in WMNs shows, the Babel routing protocol outperforms [3] OLSR and BATMAN.

For the TinyOS operating system two multi-hop routing protocols exist in the standard distribution: Tymo and BLIP. Tymo [7] is a TinyOS implementation of the reactive Dymo [5] AODV routing protocol. The Berkeley low-power IP

(BLIP) stack, is an IPv6 implementation for TinyOS. It includes IPv6 neighbour discovery, default route selection and point-to-point routing [15]. BLIP allows to form multi-hop IP networks, which can communicate over shared protocols and can be published into the public network to provide global connectivity [16].

BLIP is too heavyweight for the needs in HydroNet, because there is no need for prefix based routing and IP addresses. On the other hand Tymo implies a delay due to its reactive nature when a new route needs to be discovered and routing information is flooded across the network. Due to the promising features and performance results of Babel [4] and the need to eliminate delays in route discovery we were inspired to develop a simplified subset implementation of Babel for TinyOS. Since TinyOS has a significantly different architecture than Linux distributions and the hardware on TinyOS devices is much more limited, several simplifications and changes have to be made compared to the full set implementation.

Section II explains the basic features of Babel. Readers not familiar with DSDV [6] and Babel [1] should refer to the corresponding literature to get a better understanding of the protocols. Section III describes the simplifications and considerations made for the subset implementation in TinyOS.

During the time of writing the subset implementation has not been finished, and therefore, the paper was submitted as *Work in Progress*. Final performance results, compared to Tymo and BLIP, will follow in the future which will prove if Babel multi-hop routing performs well in the TinyOS architecture.

## II. BABEL ROUTING PROTOCOL

Babel is a loop-avoiding distance-vector routing protocol designed to run in wired and in highly dynamic wireless networks. It runs in networks using prefix-based or flat routing (mesh networks) and is able to operate with IPv4 and IPv6 protocols on multiple interfaces simultaneously. Babel puts routing information into a type-length-value (TLV) [19] format and aggregates multiple TLVs into one single packet. Optionally a Babel node can request an acknowledgment for any Babel packet it sends by adding an Acknowledgment Request TLV. A Babel node periodically broadcasts Hello TLVs to all of its neighbours; it also periodically sends an IHU (I Heard You) TLV to every neighbour from which it has recently heard a Hello [1]. From the information derived from Hello and IHU TLVs, a node calculates the cost  $c$  (from the transmission and reception cost [1, Section 3.4]) for a link to a specific neighbour. Additionally a Babel node periodically advertises its set of selected routes to its neighbours with Update TLVs. Each route contains a sequence number  $s$  and a metric  $m$  for a node  $n$ . The sequence number  $s$  determines the freshness of the route advertisement and is propagated unchanged through the

network and is only incremented by  $n$ . For example, if a node receives two route advertisements for  $n$  from two different neighbours, it will take the route with newer  $s$ . Compared to DSDV, Babel speeds up convergence when the topology changed by reactively requesting a new sequence number (with a sequence number request TLV) instead of waiting until the new sequence number is sent in the next periodic interval [1, Section 2.6]. Babel uses a feasibility condition, taken from EIGRP and less strict than AODV, that guarantees the absence of routing loops. A stable Babel routing daemon, which runs on Linux and Mac OS X, is available [2].

## III. BABEL FOR TINYOS

This section describes the main simplifications made to the full-blown Babel implementation to fit well into the TinyOS architecture and the project needs for HydroNet.

### A. No interface table

Babel specifies an interface table, which contains a list of network interfaces on which a node understands the Babel protocol [1]. Almost all supported platforms [17] for TinyOS have two interfaces, usually a RS232 and a radio interface. The robot's main controller in HydroNet is connected via RS232 to the communication infrastructure but does not participate in the routing process and therefore no interface table is needed.

### B. Flat routing

Babel is designed to run in networks using prefix-based routing [8] and in networks using flat routing. Traditional wired IP networks (prefix-based routing) have a hierarchical address space. Such an address identifies a node in the network and also provides information about the location in the hierarchical topology. Since nodes in WMNs are free to move, an address should only identify a node in such a network. HydroNet's Babel implementation for TinyOS focuses on a single WMN, which is not interconnected, like Hybrid Wireless Mesh Networks (HWMNs). For this reason, a simpler addressing scheme can be used as described in section III-C and the prefix-based routing can be omitted.

### C. Addressing

Babel is specified to run on dual-stack networks. Therefore, all Babel packets with an address field also have an address encoding field which indicates if it is a wildcard, IPv4 or IPv6 address.

TinyOS typically uses active messages (AM) [18] and the packet abstraction *message\_t* [9] for communication. The AM default address representation is an unsigned 16 bit integer but also different representations like IPv4 or IPv6 can be defined. By solely using the 16 bit AM address representation, the address encoding field can be omitted and a lot of space can be saved in messages and memory.

#### D. Fewer Babel data types

The bulk of Babel routing traffic consists of route advertisements. Since Babel runs on dual-stack networks, most of the overhead is spent on the large IPv4 and especially IPv6 addresses. However Babel uses address compression to minimize the packet size. If multiple Update TLVs in a packet share the same prefix, only the first one contains the prefix. Consecutive Update TLVs will derive the prefix from the first one. Additionally a Next Hop TLV advertises a next hop address that is implied by subsequent Update TLVs. If no Next Hop TLV is present, the next hop address is taken from the network layer source address. For 16 bit addresses Next Hop TLVs will introduce an overhead if only a few route advertisements share the same next hop address. The HydroNet implementation uses flat routing and therefore no prefixes are required. For this reasons HydroNet's implementation does not use address compression and the next hop address is carried directly in Update TLVs.

If in Babel a node receives an Acknowledgment Request TLV in a packet, it should reply with an Acknowledgement TLV within the interval specified in the request. Since Babel is designed to deal gracefully with packet loss on unreliable media HydroNet's implementation will rely on periodic updates to ensure that any usable routes are eventually propagated and therefore no acknowledgment mechanism is implemented.

Because flat routing is used and no multiple edge routers are participating for the routing domain, the Router-Id TLV is not used [1, Section 2.7].

#### E. Neighbour Discovery & Route Advertisement intervals

A node maintains an interval for Hello, IHU and Update TLVs. The interval is carried in those TLVs and specifies the time after the node will send a new TLV of that type. Therefore a receiving node can identify if a neighbour changed one of its intervals. A neighbour can increase the Hello, IHU and/or Update intervals to prevent too frequent transmissions of routing packets to reduce battery consumption at the expense of other nodes may have outdated knowledge about this particular neighbour.

Additionally a node can maintain a counter, which counts how many packets already have been forwarded to other nodes for a specific time interval. If this value is below a specified threshold, this node is most probably not participating much as a router in the whole multi-hop routing process and can increase its intervals (to send fewer Hello, IHU and Update TLVs). This means, that in a worst-case scenario, where nodes may suffer from an outdated view of this particular node, just a small amount of traffic might be affected. When the forwarded packet counter of the node increases again above the threshold, it can decrease its intervals (Hello, IHU and Update TLVs will be sent more frequently) to ensure that other nodes have a more up-to-date view of the node itself.

HydroNet's implementation relies on this mechanisms to extend battery lifetime and reduce bandwidth usage.

#### F. Metric computation

The Babel specification requires a monotonic and isotonic metric. The simplest approach will be to define a metric of a route as the sum of the costs of all component's links from the source to destination node [1]. If a neighbour advertises a route with a metric  $m$  over a link with cost  $c$ , the resulting route has a metric of  $c + m$ .

The Babel specification also allows external sources, for example the battery level or CPU load, to be taken into account of a metric. This can be achieved by adding a value  $k$  that depends on the external source of data to every route's metric. Therefore a node might compute a metric as  $k + c + m$ , where the value of  $k + c$  must be greater than 0 to preserve strict monotonicity.

The Received Signal Strength Indication (RSSI) should not be used as a source for metric. Srinivasan and Levis [21] showed that RSSI does not correlate well with the packet reception rate and that Link Quality-Based Routing metrics can provide a more accurate estimation of the link cost. The Estimated Transmission Count (ETX) [23] is a bidirectional Link Quality-Based metric computation mechanism and is also used for wireless links in the Babel daemon. ETX is similar to the Link Estimation Exchange Protocol (LEEP) [10], which is used in TinyOS. Instead of using LEEP, Hello and Update TLVs can be used to carry the information needed for the ETX computation.

#### G. Packet Format

A Babel packet consists of a 32 bytes header, followed by a sequence of one or more TLVs. The default payload size of a radio packet in TinyOS is 28 bytes [9] and due to the small size it was considered to exclude the TLV format and TLV aggregation entirely for Babel packets to minimise overhead. Communication tests identified a payload size of 144 bytes for an optimal maximum sustained throughput, which means high throughput at a minimal packet collision rate. This payload size made the use of TLV aggregation affordable again for HydroNet's implementation.

#### H. Broadcasting IHU packets

The Babel RFC states that IHUs are conceptually unicast but they should be sent to a multicast address in order to aggregate multiple IHU TLVs in a single packet. In HydroNet multicast addresses are not available. Therefore HydroNet's IHU packets are broadcast to the neighbours, containing one or more IHU TLVs, which define for which neighbours the packet is destined. When a node receives an IHU broadcast packet, it will parse the containing IHU TLVs. If no IHU TLV is addressed for the node, it will silently ignore the broadcast. This technique has a similar effect as multicast and it is not required to send multiple unicast IHU packets.

#### IV. CONCLUSION

With all the mentioned simplifications and changes we were able to integrate a Babel subset implementation into our resource constrained hardware. The routing exchange information fits well, with aggregation of multiple TLVs, into a 144 bytes packet and therefore no fragmentation occurs. Instead of a node sending multiple IHU packets to all of its neighbours it can broadcast one IHU packet (see section III-H) which leads to fewer IHU packets needed to be sent. Final results will follow in the future when the performance tests are done, which will show if HydroNet's Babel implementation performs better in the TinyOS landscape than other existing TinyOS multi-hop routing protocols.

#### ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2011] under grant agreement no. 212790.

Special thanks goes to Juliusz Chroboczek, the inventor of Babel, who explained us his routing protocol and answered our questions patiently.

#### REFERENCES

- [1] J. Chroboczek, *The Babel Routing Protocol*, RFC 6126, ISSN 2070-1721, April 2011.
- [2] J. Chroboczek, *Babel routing daemon*, <http://www.pps.jussieu.fr/~jch/software/babel/>. (Accessed 13. September 2011).
- [3] D. Murray, M. Dixon and T. Koziniec, *An Experimental Comparison of Routing Protocols in Multi Hop Ad Hoc Networks* in The Australian Telecommunication Networks and Applications Conference 2010, p. 162, 2010.
- [4] M. Abolhasan, B. Hagelstein and J. Wang, *Real-world Performance of Current Proactive Multi-hop Mesh Protocols* in Asia-Pacific Conference on Communication 2009, p. 5, 2009.
- [5] I. Chakeres and C. Perkins, *Dynamic MANET On-demand (DYMO) Routing*, Internet-Draft, July 2010.
- [6] C. Perkins and P. Bhagwat, *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers* in Special Interest Group on Data Communication 1994, pp. 234-244, October 1994.
- [7] R. Thouvenin, *Implementing and Evaluating the Dynamic Manet On-demand Protocol in Wireless Sensor Networks*, Master's thesis, University of Aarhus Department of Computer Science, 2007.
- [8] V. Fuller, T. Li, J. Yu and K. Varadhan, *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*, RFC 1519, September 1993.
- [9] P. Levis, *message\_t*, TinyOS Extension Proposal 111, <http://www.tinyos.net/tinyos-2.x/doc/html/tep111.html>, August 2007. (Accessed 8. September 2011).
- [10] O. Gnawali, *The Link Estimation Exchange Protocol (LEEP)*, TinyOS Extension Proposal 124, <http://www.tinyos.net/tinyos-2.x/doc/html/tep124.html>, February 2007, rev. 1.4. (Accessed 13. September 2011).
- [11] Tinynode 584, *Tinynode 584 Fact Sheet*, [http://www.tinynode.com/?q=system/files/TN584\\_Fact\\_Sheet\\_v\\_1\\_1\\_1.pdf](http://www.tinynode.com/?q=system/files/TN584_Fact_Sheet_v_1_1_1.pdf). (Accessed 15. September 2011).
- [12] G. Malkin, *RIP Version 2*, RFC 2453, November 1998.
- [13] J. Moy, *OSPF Version 2*, RFC 2328, April 1998.
- [14] P. Jacquet, A. Laouiti, P. Minet and L. Viennot, *Performance of Multipoint Relaying in Ad Hoc Mobile Routing Protocols*, in NETWORKING 02: Proceedings of the Second International IFIP-TC6 Networking Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; and Mobile and Wireless Communications, London, UK, 2002, pp. 387-398, Springer-Verlag.
- [15] Berkeley Wireless Embedded Systems. *Berkeley IP Information*, <http://smote.cs.berkeley.edu:8000/tracenv/wiki/blip>. (Accessed 2. September 2011).
- [16] TinyOS Tutorials. *BLIP Tutorial*, [http://docs.tinyos.net/tinywiki/index.php/BLIP\\_Tutorial](http://docs.tinyos.net/tinywiki/index.php/BLIP_Tutorial). (Accessed 31. October 2011).
- [17] Hardware designs. *Hardware supported by TinyOS*, <http://www.tinyos.net/scoop/special/hardware>. (Accessed 25. August 2011).
- [18] P. Buonadonna, J. Hill and D. Culler, *Active Message Communication for Tiny Networked Sensors*, September 2000.
- [19] ISO/IEC 7816-4, *Identification cards - Integrated circuit cards*, Part 4: Organization, security and commands for interchange, January 2005, Second edition, p. 13, International Organization for Standardization.
- [20] S. Floyd and V. Jacobson, *The synchronization of periodic routing messages*, IEEE/ACM Transactions on Networking, pp. 122-136, April 1994.
- [21] K. Srinivasan and P. Levis, *RSSI is Under Appreciated*, in Proceedings of the Third Workshop on Embedded Networked Sensors, p. 3, 2006.
- [22] HydroNet, *Objectives*, <http://www.hydro-net-project.eu/index.php?menu=objectives>. (Accessed 17. August 2011).
- [23] D. De Couto, *High-Throughput Routing for Multi-Hop Wireless Networks*, p. 55, June 2004.