6LoWDTN: IPv6-Enabled Delay-Tolerant WSNs for Contiki

Luca Bruno, Mirko Franceschinis, Claudio Pastrone, Riccardo Tomasi, Maurizio Spirito
Pervasive Technologies Area
Istituto Superiore Mario Boella
Torino, Italy
Email: lucab@unstable.it, franceschinis@ismb.it, pastrone@ismb.it, tomasi@ismb.it, spirito@ismb.it

***Abstract*—Low-power Wireless Personal Area Networks (LoWPANs) solutions such as Wireless Sensor Networks (WSNs) can be deployed in scenarios characterized by partial node mobility and unreliable radio links. In such challenging conditions, normal ad-hoc networking approaches might result in unstable topologies and major data losses, thus hindering or limiting the validity of the solution. In case the application does not entail delay requirements, Delay Tolerant Networking (DTN) approaches could improve data delivery, thanks to cooperative nodes selectively storing and forwarding data to counter bad connectivity. In order to cope with such issues, this paper proposes 6LoWDTN, a DTN transport layer for WSNs built on-top the 6LoWPAN stack and derived from CHARON opportunistic networking solution. 6LoWDTN has been designed and implemented based on the μIPv6 stack provided by the Contiki operating system.**

***Index Terms*—WSN, DTN, 6LoWPAN, IPv6, CHARON**

I. Introduction

Low-power Wireless Personal Area Network (LoWPAN) are self-configuring ad-hoc networks composed of wirelessly-connected autonomous devices, usually characterized by constrained computational and memory resources and low consumption. Wireless Sensor Networks (WSNs) are currently the most common type of LoWPAN. According to the WSN paradigm, devices are designed to collaboratively sense the environment and communicate with each other in a multi-hop fashion. WSNs represent a practical approach to perform distributed long-term monitoring tasks in large-scale environments and thus cover a key role in enriching any ICT system with information from the "field".
According to the most recent developments, WSNs are gradually migrating towards an Internet-connected paradigm, in order to ensure easier flexibility and integration with existing Internet-enabled solutions. Such paradigm assumes that any tiny WSN device could be considered a fully-compliant Internet node [1]. Due to the expected large number of interconnected devices the IPv6 [2] standard is considered a key technology to be adopted in this context. Nevertheless, IPv6 can not be employed "as-is" in WSNs and thus adaptation layers such as IPv6 over LoWPAN, known as 6LoWPAN [3], [4], [5], are under investigation and standardization.
Beyond adaptation issues (which are not negligible), networking in the proposed environment can be hindered by a number of communication challenges, which might result in poor performance if unreliable communication approaches are adopted. Firstly, LoWPANs might operate in harsh (lossy) communication conditions, due to bad propagation environments, interference from devices employing more powerful radio technologies working in the same frequency band (e.g. WiFi), EMC issues or even intentional jamming. Secondly, LoWPAN nodes might be not always available, due to radio or MCU sleep cycles employed to minimize energy consumption. Finally, in some scenarios a number of LoWPAN nodes might operate in mobility conditions.
As a result, LoWPAN communication might be characterized by intermittent, unreliable links and physical network topologies which vary even significantly over time, as summarized by the reference scenario described in Fig. 1.
A number of Border Routers (BRs), i.e. edge nodes, provide the bi-directional interface between the wired Internet and the LoWPAN, which is composed both of Static Nodes (SNs), dwelling in fixed places, and Mobile Nodes (MNs), whose position changes over time. Whereas possible, SNs extend the radio coverage of BRs in a mesh fashion, providing connectivity to other SNs or MNs through a multi-hop approach.

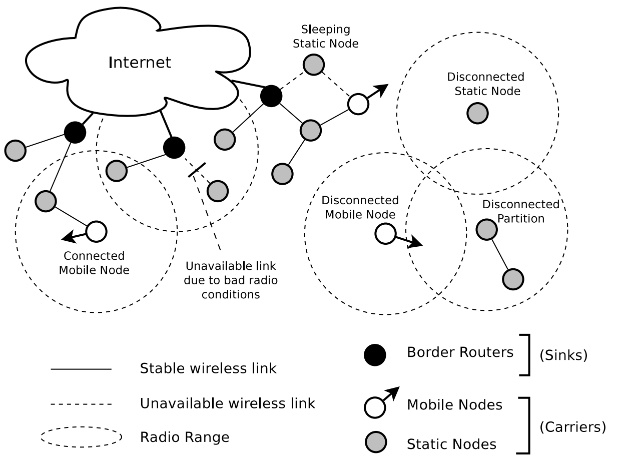

Fig. 1. Reference Scenario

Acknowledgements: The authors gratefully acknowledge partial funding of this work by the regional project "Piattaforma Tecnologica Innovativa per l'Internet of Things", and the collaboration between ISMB and Politecnico di Torino within the FP7 Network of Excellence Newcom++ (contract no. 216715).

As MNs move, they can be either in-coverage (connected) or out of coverage (disconnected). Occasionally, as the topology cannot be always planned in advance and radio links are not always reliable, also SNs, or even groups of SNs (i.e. a partition), might be disconnected from the remaining of the network.

In such a scenario, normal ad-hoc networking approaches might be not always efficient, resulting in loss of data from a possibly significant part of the network. In applications where delay is not an issue (i.e. there are no requirements concerning the minimum time for data delivery from the source to the destination) it is considered acceptable for the nodes to temporarily store data and wait for better connectivity conditions and more valuable forwarding opportunities. The related "store-carry-and-forward" paradigm is the basic principle of Delay Tolerant Networks (DTNs)[6].

DTN architectures offer an "overlay" infrastructure which takes advantage of existing data transports to carry "bundles" of data from one end to the other of the network, providing primitives to perform persistent storing and occasional forwarding of bundles as well as a simple mechanism to select a known-good path toward the target destination. A "carrier" can sense data from the environment, store it until more favourable conditions arise, carry it while moving in the area, forward it to another (hopefully better positioned) carrier or directly to the final destination, also called "sink". As shown in Fig. 1 carriers can be any MN and SN, while sinks are normally colocated with BRs.

Since DTNs have been designed for occasionally-connected networks that may suffer from frequent partitions, including total network disruption, and high-delays, they provide a promising approach in many scenarios where WSNs and LoWPANs are applied in harsh environments. In order to better investigate its applicability and support further research in this area, this paper proposes 6LoWDTN, a DTN transport layer positioned on-top of a 6LoWPAN stack, which has been designed and implemented based on the Contiki operating system [7]. Proposing a real implementation of an opportunistic routing approach relying on a 6LoWPAN layer, this paper wants to open the way to forthcoming experimentations of existing WPAN platforms in the IP world under as challenging as expectable scenarios of the near future.

The remaining of the paper is structured as follows: Sec. II provides a taxonomy of background and related work; Sec. III describes the 6LoWDTN reference design, with implementation notes in Sec. III-A; finally, Sec. IV draws conclusions and outlines on-going and future work.

II. Background

As discussed in the previous section, packet forwarding is the strategy commonly adopted in multi-hop ad-hoc wireless networks in order to enable the communication between couples of nodes when their positions are such that one is not in the other's communication range. Deployments over large areas and low node density represent contexts in which the need for forwarding schemes is further emphasized.

In presence of mobile nodes the scenario is even more critical since the distance between any two nodes is time-varying, as well as the link characterization and the possibility of a direct point-to-point communication. Nonetheless node mobility can be favorably exploited to carry data across the network, towards the final destination, featuring an opportunistic networking approach. New challenging aspects arise in the decision whether (and possibly to whom) forwarding data in case of contact events.

Significant research efforts on DTNs are actively pursued by the DTN Research Group within Internet Research Task Force (IRTF). Several informational and experimental specifications already describe characteristics and protocols for DTN operating on top of existing Internet Protocols. Drafted documents currently include DTN architectural overview [8], bundle packet format (Bundle Protocol [9]), transmission protocol (LTP [10]) and security extensions [11].

Beyond IRTF efforts, different approaches exist in the literature facing the problem of opportunistic networking, and, more specifically, opportunistic routing and protocols. The simplest though less efficient schemes, in particular in high traffic intensity applications, are based on flooding: the unsophisticated Epidemic Routing [12] and the more advanced Spray and Wait [13], which adds controls to avoid unnecessary duplications of messages, fall in this category. Unlike flooding-based protocols, other proposals keep a unique copy of the message in the network. However, they heavily differ by other aspects. MULE [14] introduces special nodes with well-defined resources, roles and tasks: mere sensors, mobile agents and access points. Mobile agents, called Data Mules, are always in movement, collecting data from distributed sensors and delivering it to sinks. When some correlation is expected to subsist between past, present and future node mobility patterns, history-based protocols like PROPHET [15] and SCAR [16] can exhibit good performance. The criteria for forwarding message copies in the network are someway derived by statistics concerning previous inter-node contacts.

CHARON (the reader can refer to [17], [18] for major details) can be considered to offer a hybrid approach. As in flooding-based protocols, it admits the existence of multiple copies by introducing the concept of "zombie" copies. However, if and when desired, zombie copies can be erased such to make CHARON a single-copy protocol. Zombie copies have no additional energy cost since they are the product of inevitable copying occurring when a CHARON node forwards a message. Instead of immediately deleting the copy, the forwarding node keeps it in the buffer until it is convenient and its suppression can be decided at any moment, typically when memory space is needed. Zombie copies cannot be forwarded, but only directly delivered to a sink, when met. Finally, it is worth noting that CHARON can also support a multi-copy mode similar to PROPHET's in those situations in which a specific message is required to arrive to the sink quickly.

Although introduced as an opportunistic forwarding scheme, CHARON is much more: it can be seen as a complete opportunistic solution defining specifications at multiple layers, included synchronization service, energy management, quality of service and application data traffic generation.

In CHARON, time synchronization is employed both for network overlay maintenance and for bundles meta-data. Time reference broadcast by the sink is subsequently propagated by carriers in order to achieve shared time-reference across the network. Bundle time-stamping is carried out by each node originating data messages, as the sink relies on it to reconstruct the chronological order of received bundles. Time synchronization in CHARON is not as extremely accurate as in other popular protocols for WSNs [19], [20], [21], [22] but it is very simple and able to guarantee the management of global radio duty-cycles on which energy saving is based.

In a CHARON network, nodes are at the same time data message generators and potential message carriers towards recipient nodes called sinks. When two nodes enter in contact, i.e. they can directly communicate, each one determines if it is preferable to forward currently held messages to the potential carrier or to continue storing them. The decision is taken based on the calculation of a utility function, defined as a weighted sum of a set of metrics. The weights can be tuned as desired and the set of metrics can include the inter-contact time and the estimated delivery delay, which are the most relevant, as well as, for instance, battery level and free buffer space. In order to make nodes aware of contacts, during the activity periods they periodically exchange beacon messages, sent in broadcast, that contain pieces of information of various nature (current battery level, inter-contact times, estimated delivery delays, synchronization data, etc.).

Quality of service (QoS) is not supported at connection level, i.e. all classes are provided with best-effort service. Instead, QoS is goal-based, since the user is let the chance of choosing the application objective, for instance minimize latency or maximize energy efficiency. As previously mentioned, such flexibility is implemented in the utility function where weights can be set as preferable.

Definitively, CHARON appears as a flexible solution primarily in virtue of allowing the tuning of a large set of parameters which, depending on application goals and constraints, can give priority to simplicity, configuring with minimal capabilities and complexity, or can combine multiple information data sources. For these reasons, CHARON has been adopted as the enabling bundle layer protocol, offering primitives for time synchronization, path selection, store-carry-forward of bundles as well as custody transfer. Compared to the IRTF architecture, the adopted approach maintains most of the premises described in the general DTN architecture specification[23]. As IRTF efforts are mainly focused on interplanetary and deep space networks, strictly opportunistic behavior and typical LoWPAN requirements (ie. constrained resources and small-sized frames) are not considered primal in that context. For such reasons, a different bundle format and forwarding algorithm has been adopted in this paper. Security features are not currently defined within this architecture, but can be derived from currently drafted bundle security specifications[11], once officially formalized.

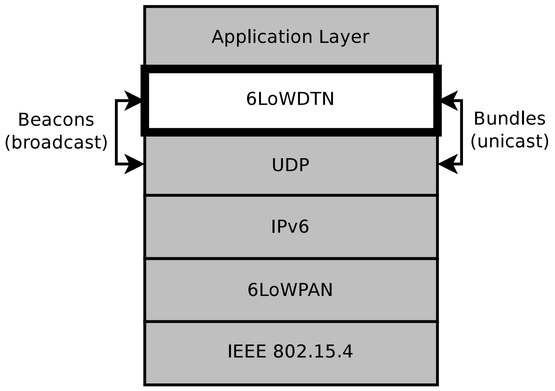

Fig. 2. 6LoWDTN complete network stack

III. 6LoWDTN Design

6LoWDTN is a bundle layer providing transport functionalities to applications, leveraging on UDP for communication, as shown in Fig. 2. Although in strict ISO/OSI terms 6LoWDTN is a transport layer just as UDP, the latter has been kept in the stack for easier integration with existing applications based on best-effort packet-oriented communication. The underlying IPv6/6LoWPAN layer is also employed for node addressing.
6LoWDTN introduces additional overlay routing, forwarding and synchronization on top of the underlying layers.
The adopted overlay routing algorithm is in charge of deciding whether to forward a bundle or not to a neighboring node. As in CHARON, such decision is based on the evaluation of a metric named Estimated Delivery Delay (EDD), moderated by a score result provided by a utility function. The EDD is an estimation of the total delay required to transport a bundle from the local node to its final destination sink; the EDD is locally computed by each node, based on its history of contacts with other nodes. The utility function, instead, is used to take into consideration in the routing decision process heterogeneous decision metrics (eg. local buffer occupation, residual battery power,…), for sake of global network optimization. Such approach has proven to be both flexible and efficient, and is currently in use also in other 6LoWPAN-related routing algorithms. For example, RPL [24] makes large use of Object Functions to select the optimum routing path, with the possibility of adapting to different network conditions and parameters just using different functions.
In order to keep EDD and other metrics up to date, nodes can exchange link-local broadcast UDP beacons. Such beacons are employed to exchange routing metrics among neighboring nodes. Beacons are periodically sent in order to signal each node reachability, possibly triggering bundle forwarding from adjacent nodes.
The structure of beacons is shown in Fig. 3: *TimeStamp* is the value of the local clock, propagated for time synchronization purposes; *TimeAge* indicates the “freshness” of the *TimeStamp* field i.e. time since the last update of the local clock with a reliable time source; *Score* and *EDD* contain the value

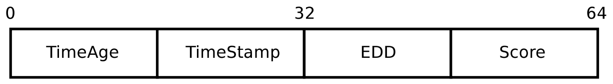

Fig. 3. Beacon packet structure

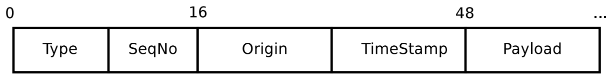

Fig. 4. Bundle header structure

of aforementioned metrics computed by the local node. The primal source of time synchronization information are sink nodes, whose *TimeAge* and *EDD* values are always zero. Carrier nodes instead are able to synchronize their timestamp with references acquired through neighbors.

Bundle forwarding employes point-to-point communication in the form of UDP unicast packets. Sink nodes are the final destination of bundles, while carriers act both as data sources and as intermediate forwarding points toward sinks. Available bundles are forwarded upon each inter-node contact, if conditions enforced by the overlay routing algorithm are met.

UDP bundles usually carry both the application level payload and a minimal set of bundle layer headers, strictly linked to each bundle. The structure of these additional bundle headers is shown in Fig. 4: *Type* defines the priority assigned to the bundle; *SeqNo* marks sequentially each bundle generated by the local node; *Origin* indicates the IPv6 address of the originating node; *TimeStamp* marks the current value of the clock of the originating node at the moment when data is generated; *Payload* contains the actual contents of the bundle. The bundle priority defined by the *Type* field has an impact on both the forwarding and local storing policy adopted by intermediate nodes, as urgent bundles result in higher priority transmission and reduced on-board storing life-time.

Upon the complete transmission of a bundle, i.e. as soon as its acknowledgment by a neighboring has been received, its local copy is not immediately deleted, but marked as Zombie. Zombie copies are locally kept on a persistent storage device, such as internal flash memory, as long as enough space for fresh copies is available. If a node eventually moves within the physical reception range of the sink, Zombie copies will be directly transferred in order to decrease bundle forwarding latency without wasting excessively node resources.

A. Proposed implementation

6LoWDTN has been implemented within the Contiki operating system [7] as a proof of work and deployed within a preliminary experimental testbed. In addition, a sample 6LoWDTN-enabled test application generating fake data has been developed to validate the solution.

The implementation has been tested using Memsic Telos rev. B nodes[25], which are composed of a low power TI MSP430 microcontroller, 1024KB of external flash for data logging and

TABLE I
ROM OCCUPATION (BYTES)

Section	6LoWDTN + μIPv6	μIPv6 only	6LoWDTN only (overhead)
.text	34910	30398	4512
.data	216	148	68
.bss	7812	7808	4
.vectors	32	32	0
Total	42970	38386	4584

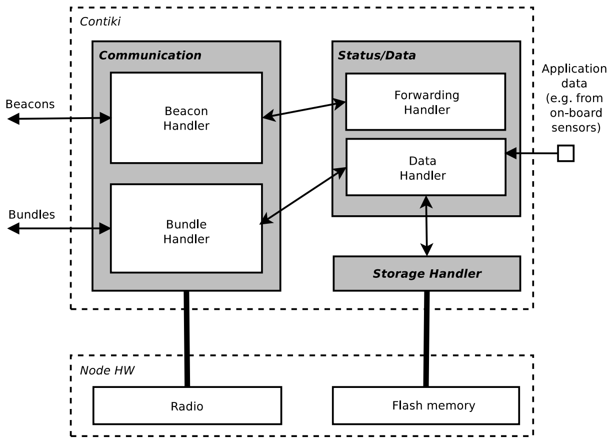

Fig. 5. Reference implementation architecture overview

a CC2420 radio. Such hardware configuration provides a good energy/computational power ratio, a persistent storage space and a IEEE 802.15.4 capable radio.

At the same time, nodes hardware poses some constraints with regard to code size, RAM occupation and sustained CPU utilization. In order to meet these constraints, proposed implementation takes advantage of the compact 6LoWPAN stack provided by Contiki, as well as the existing power management architecture.

Non crucial μIPv6 modules (e.g. TCP support) has been left out from the 6LoWDTN stack, achieving a ROM size comparable to a barebone μIPv6 application, as shown in Table I.

Aforementioned implementation choices led to a final architecture which is process-based and event-driven, organized in three major blocks as shown in Fig. 5.

The *Communication* block wraps all the relevant network primitives exposed by the Operating System, based on the UDP support from the Contiki μIPv6 stack [26]. The proposed implementation exploits the 6LoWPAN adaptation layer to achieve UDP/IPv6 packet transmission on a IEEE 802.15.4 based network. Communication and Status/Data processes are decoupled in order to better exploit the Contiki parallel protothread scheduling. Correct flow of execution is achieved through events posting between involved processes.

The *Beacon Handler* is in charge of broadcasting periodic beacons on the network, as well as receiving and processing beacons from neighbor nodes. Beacon sending is triggered by the OS, by means of a system timer which posts a specific event for this process. Neighbors information extracted from received beacons are posted to the Status process (to whom EDD refreshing is demanded). Beacons reception also activates further bundle forwarding, whenever the node has pending data yet to transmit.

The *Bundle Handler* is in charge of deciding which bundles must be forwarded, and eventually handling their unicast transmission. The receiving node takes care of acknowledging each bundle with its Sequence Number, in order to let the sender mark its local copy as forwarded. Status information exposed by the Status and Data module are exploited to decide if any bundle has to be transmitted, and to select the most appropriate one according to algorithms described in previous sections (i.e. urgent vs. normal bundles, fresh vs. zombie bundles for the sink). Payload and meta-data retrieving is demanded to the *Data Handler*, which interacts with the bundle forwarder for the actual transmission. Data Handler is again involved upon acknowledgment reception, in order to remove or zombify successfully transmitted bundles. The *Storage Handler* provides simple storage capabilities based on the Contiki COFFEE File System (CFS)[27]. CFS offers a barebone file-based structure and simplified POSIX-like APIs to store, access and handle data on persistent storage. In particular, on-board flash memory may be formatted and used as a CFS root device to store up to 1024kB of data.

The Storage Handler relies on CFS primitives to store and later retrieve bundles in their path towards the sink. Bundles are sequentially stored on the FS as separated files, containing both bundle payload and meta-data. Bundle files are organized in a flat hierarchy and placed in the root directory, avoiding distinctions based on freshness or priority. Dedicated list structures are in use to lookup bundles and filenames, subsequently categorizing fresh, urgent and zombie bundles for quick insertion and extraction. Bundles acquired locally or from other nodes are stored as fresh, which are then demoted to zombies once forwarded to another carrier. Storage process keeps zombie bundles on local FS as long as enough space is available to acquire new fresh bundles. Once the FS is filled beyond threshold, bundles to be eliminated are selected among old zombies and expired urgent bundles.

The *Status/Data* block is in charge of keeping up-to-date information about its own state, including estimated forwarding delay, remaining battery power, storage occupation and time synchronization. The Status/Data block provides the entry interface for application data, e.g. environmental data from on-board sensors, external probes, connected as I/O devices or dedicated remote sensing end-points. Functionalities provided by this module are triggered either by events posted by other processes (e.g. upon beacon or bundle reception) or by system timers (e.g. for beacon broadcasting).

Finally, the *Forwarding Handler* implements core CHARON algorithms to calculate the EDD and select appropriate next hop for bundle forwarding. This module is also in charge of computing Score values through a hard-coded utility function.

IV. Conclusions

This paper introduced 6LoWDTN, a proposal of DTN layer implementation derived from CHARON and built on-top of the 6LoWPAN stack of the Contiki OS, describing its reference architecture, protocols and the main internal features.

The main goal of 6LoWDTN is to increase reliability of LoWPAN and WSN solutions deployed in harsh environment characterized by intermittent connectivity and nodes mobility. While so far we have deployed a basic experimental testbed and developed a simple test application to validate the solution, future work will focus on extensive 6LoWDTN performance evaluation within actual application scenarios.

References

[1] A. Dunkels and J. Vasseur, "IP for Smart Objects," Internet Protocol for Smart Objects Alliance, IPSO White Paper 1, 2008.

[2] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, RFC 2460 (Draft Standard), Internet Engineering Task Force Std. 2460, Dec. 1998, updated by RFCs 5095, 5722, 5871. [Online]. Available: http://www.ietf.org/rfc/rfc2460.txt

[3] N. Kushalnagar, G. Montenegro, and C. Schumacher, *IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals*, RFC 4919 (Informational), Internet Engineering Task Force Std. 4919, Aug. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4919.txt

[4] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, *Transmission of IPv6 Packets over IEEE 802.15.4 Networks*, RFC 4944 (Proposed Standard), Internet Engineering Task Force Std. 4944, Sep. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4944.txt

[5] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet (Wiley Series on Communications Networking & Distributed Systems)*, Wiley, Ed. Wiley, 2010.

[6] Jain, Sushant and Fall, Kevin and Patra, Rabin, "Routing in a delay tolerant network," *SIGCOMM Comput. Commun. Rev.*, vol. 34, pp. 145–158, August 2004.

[7] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, nov. 2004, pp. 455 – 462.

[8] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-Tolerant Networking Architecture," RFC 4838 (Informational), Internet Engineering Task Force, Apr. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4838.txt

[9] K. Scott and S. Burleigh, "Bundle Protocol Specification," RFC 5050 (Experimental), Internet Engineering Task Force, Nov. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5050.txt

[10] M. Ramadas, S. Burleigh, and S. Farrell, "Licklider Transmission Protocol - Specification," RFC 5326 (Experimental), Internet Engineering Task Force, Sep. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5326.txt

[11] S. Symington, "Bundle Security Protocol Specification," Internet Research Task Force, Internet-Draft draft-irtf-dtnrg-bundle-security-19, 2011, work in progress. [Online]. Available: http://tools.ietf.org/html/draft-irtf-dtnrg-bundle-security-19

[12] Amin Vahdat and David Becker, "Epidemic routing for partially-connected ad hoc networks," Tech. Rep., 2000.

[13] Spyropoulos, Thrasyvoulos and Psounis, Konstantinos and Raghavendra, Cauligi S., "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, ser. WDTN '05. New York, NY, USA: ACM, 2005, pp. 252–259.

[14] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: modeling and analysis of a three-tier architecture for sparse sensor networks," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 215 – 233, 2003, sensor Network Protocols and Applications.

[15] Lindgren, Anders and Doria, Avri and Schelén, Olov, "Probabilistic Routing in Intermittently Connected Networks," in *Service Assurance with Partial and Intermittent Resources*, ser. Lecture Notes in Computer Science, Dini, Petre and Lorenz, Pascal and de Souza, José, Ed. Springer Berlin / Heidelberg, 2004, vol. 3126, pp. 239–254.

[16] Pasztor, B. and Musolesi, M. and Mascolo, C., “Opportunistic mobile sensor data collection with scar,” in *MASS 2007. IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems, 2007.* .

[17] Soares, J.M. and Rocha, R.M., “CHARON: Routing in low-density opportunistic wireless sensor networks,” in *Wireless Days (WD), 2009 2nd IFIP*, dec. 2009, pp. 1 –5.

[18] Soares, Jorge M. and Franceschinis, Mirko and Rocha, Rui M. and Zhang, Wansheng and Spirito, Maurizio A., “Opportunistic data collection in sparse wireless sensor networks,” *EURASIP J. Wirel. Commun. Netw.*, vol. 2011, pp. 6:1–6:20, January 2011.

[19] Maróti, Miklós and Kusy, Branislav and Simon, Gyula and Lédeczi, Ákos, “The flooding time synchronization protocol,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ser. SenSys ’04. New York, NY, USA: ACM, 2004, pp. 39–49.

[20] Ganeriwal, Saurabh and Kumar, Ram and Srivastava, Mani B., “Timing-sync protocol for sensor networks,” in *Proceedings of the 1st international conference on Embedded networked sensor systems*, ser. SenSys ’03. New York, NY, USA: ACM, 2003, pp. 138–149.

[21] Fabrizio Sellone and Hussein Khaleel and Marco Urso and Mirko Franceschinis and Marina Mondin, “Accuracy-Driven Synchronization Protocol,” *Sensor Technologies and Applications, International Conference on*, vol. 0, pp. 93–98, 2008.

[22] Khaleel, Hussein and Franceschinis, Mirko and Tomasi, Riccardo and Mondin, Marina, “Accuracy-driven synchronization protocol: implementation and experimental evaluation,” in *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, ser. MEDES ’09. New York, NY, USA: ACM, 2009, pp. 55:369–55:376.

[23] Fall, K. and Farrell, S., “DTN: an architectural retrospective,” *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 5, pp. 828 –836, june 2008.

[24] T. Winter and P. Thubert, *RPL: IPv6 Routing Protocol for Low power and Lossy Networks*, Internet-Draft (work in progress), draft-ietf-roll-rpl-19, Internet Engineering Task Force Std., 2011. [Online]. Available: http://tools.ietf.org/html/draft-ietf-roll-rpl-19

[25] Memsic, *Telos rev.B datasheet*, Accessed Dec. 2010, available at http://www.memsic.com/products/- wireless-sensor-networks/wireless-modules.html.

[26] M. Durvy, J. Abeillé, P. Wetterwald, C. O’Flynn, B. Leverett, E. Gnoske, M. Vidales, G. Mulligan, N. Tsiftes, N. Finne, and A. Dunkels, “Making sensor networks ipv6 ready,” in *Proceedings of the Sixth ACM Conference on Networked Embedded Sensor Systems (ACM SenSys 2008), poster session*, Raleigh, North Carolina, USA, Nov. 2008, best poster award. [Online]. Available: http://www.sics.se/ adam/durvy08making.pdf

[27] N. Tsiftes, A. Dunkels, Z. He, and T. Voigt, “Enabling Large-Scale Storage in Sensor Networks with the Coffee File System,” in *Proceedings of the 8th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN 2009)*, San Francisco, USA, Apr. 2009. [Online]. Available: http://www.sics.se/ adam/tsiftes09enabling.pdf