A Survey of Transport Protocols for Wireless Sensor Networks

Chonggang Wang and Kazem Sohraby, University of Arkansas
Bo Li, The Hong Kong University of Science and Technology
Mahmoud Daneshmand, AT&T Labs Research
Yueming Hu, South China Agricultural University

Abstract

In this article we present a survey of transport protocols for Wireless Sensor Networks (WSNs). We first highlight several unique aspects of WSNs, and describe the basic design criteria and challenges of transport protocols, including energy-efficiency, quality of service, reliability, and congestion control. We then provide a summary and comparison of existing transport protocols for WSNs. Finally, we discuss several open research problems.

Wireless sensor networks (WSNs) generally consist of one or more sinks (or base stations) and perhaps tens or thousands of sensor nodes scattered in a physical space. With integration of information sensing, computation, and wireless communication, the sensor nodes can sense physical information, process crude information, and report them to the sink. The sink in turn queries the sensor nodes for information. WSNs have several distinctive features:

- *Unique network topology*: Sensor nodes are generally organized in a multihop star-tree topology that is either flat or hierarchical. The sink at the root of the tree is responsible for data collection and relaying to external networks. This topology can be dynamic due to the time-varying link condition and node variation.
- *Diverse applications*: WSNs may be used in different environments supporting diverse applications, from habitat monitoring and target tracking to security surveillance and so on. These applications may be focused on different sensory data and therefore impose different requirements in terms of quality of service (QoS) and reliability.
- *Traffic characteristics*: In WSNs, the primary traffic is in the upstream direction from the sensor nodes to the sink, although the sink may occasionally generate certain downstream traffic for the purposes of query and control. In the upstream this is a many-to-one type of communication. Depending on specific applications, the delivery of upstream traffic may be event-driven, continuous delivery, query-driven delivery, or hybrid delivery.
- *Resource constraints*: Sensor nodes have limited resources, including low computational capability, small memory, low wireless communication bandwidth, and a limited, usually nonrechargeable battery.
- *Small message size*: Messages in sensor networks usually have a small size compared with the existing networks. As a result, there is usually no concept of segmentation in most applications in WSNs.

These distinctive features pose new challenges in the design of WSNs that should meet application requirements and operate for the longest possible period of time. Specifically, one needs to carefully cope with such problems as energy conservation, reliability, and QoS.

Our major focus in this article is on the design of transport protocols for WSNs. Transport protocols are used to mitigate congestion and reduce packet loss, to provide fairness in bandwidth allocation, and to guarantee end-to-end reliability. However, the traditional transport protocols that are currently used for the Internet (i.e., UDP and TCP) cannot be directly implemented for WSNs [1]. For example, it is well documented that UDP does not provide delivery reliability that is often needed for many sensor applications, nor does it offer flow and congestion control that can lead to packet loss and unnecessary energy consumption. On the other hand, TCP has several other drawbacks:

- Overhead associated with TCP connection establishment might not be justified for data collection in most event-driven applications.
- Flow and congestion control mechanisms in TCP can discriminate against sensor node(s) that are far away from the sink, and result in unfair bandwidth allocation and data collections.
- It is well known that TCP has a degraded throughput in wireless systems, especially in situations with a high-packet-loss rate because TCP assumes that packet loss is due to congestion and triggers rate reduction whenever packet loss is detected.
- In contrast to hop-by-hop control, end-to-end congestion control in TCP has a tardy response, which means that it requires a longer time to mitigate congestion and in turn leads to higher packet loss when congestion occurs.
- TCP relies on end-to-end retransmission to provide reliable data transport, which consumes more energy and bandwidth than hop-by-hop retransmission.
- TCP guarantees successful transmission of packets, which is not always necessary for event-driven applications in sensor networks.

This article summarizes the features of the existing transport protocols for WSNs and outlines several open problems.

0890-8044/06/$20.00 © 2006 IEEE

The remainder of the article is organized as follows. We introduce basic principles and design criteria in transport protocols for WSNs. Next, we present a summary and comparison of existing transport protocols for WSNs. After that, we outline a list of factors that should be further considered in the design of transport protocols, and directions for further studies.

Design Guidelines for Transport Protocols in WSNS

The transport protocol runs over the network layer. It enables end-to-end message transmission, where messages may be fragmented into several segments at the transmitter and reassembled at the receiver. This protocol provides the following functions: orderly transmission, flow and congestion control, loss recovery, and possibly QoS guarantees such as timing and fairness. In WSNs several new factors, such as the convergent nature of upstream traffic and limited wireless bandwidth, can result in congestion. Congestion impacts normal data exchange and may lead to packet loss. In addition, wireless channel introduces packet loss due to bit-error rate, which not only affects reliability, but also wastes energy. As a result, two major problems that WSN transport protocols need to cope with are congestion and packet loss. We next discuss the performance metrics and required functions of transport protocols for WSNs, and discuss design options.

Performance Metrics

Transport protocols for WSNs should provide end-to-end reliability and end-to-end QoS in an energy-efficient manner. Performance of transport protocols for WSNs can be evaluated using metrics such as energy efficiency, reliability, QoS (e.g., packet-loss ratio, packet-delivery latency), and fairness.

Energy Efficiency — Sensor nodes have limited energy. As a result, it is important for the transport protocols to maintain high energy efficiency in order to maximize system lifetime. Packet loss in WSNs can be common due to bit error and/or congestion. For loss-sensitive applications, packet loss leads to retransmission and the inevitable consumption of additional battery power. Therefore, several factors need to be carefully considered, including the number of packet retransmissions, the distance (e.g., hop) for each retransmission, and the overhead associated with control messages.

Reliability — Reliability in WSNs can be classified into the following categories:

- *Packet reliability*: Applications are loss-sensitive and require successful transmission of all packets or at a certain success ratio.
- *Event reliability* [2]: Applications require only successful event detection, but not successful transmission of all packets.

In addition, [3] defines destination-related reliability. For example, messages might need to be delivered to sensor nodes in a specific subarea or to the nodes that cover a specific subarea, or to nodes that are equipped with a particular sensor type.

QoS Metrics — QoS metrics include bandwidth, latency or delay, and packet-loss ratio. Depending on the application, these metrics or their variants could be used for WSNs. For example, sensor nodes may be used to transmit continuous images for target tracking. These nodes generate high-speed data streams and require higher bandwidth than most event-based applications. For a delay-sensitive application, WSNs may also require timely delivery data.

Fairness — Sensor nodes are scattered in a geographical area. Due to the many-to-one convergent nature of upstream traffic, it is difficult for sensor nodes that are far away from the sink to transmit data. Therefore, transport protocols need to allocate bandwidth fairly among all sensor nodes so that the sink can obtain a fair amount of data from all the sensor nodes.

Congestion Control

There are mainly two causes for congestion in WSNs. The first is due to the packet-arrival rate exceeding the packet-service rate. This is more likely to occur at sensor nodes close to the sink, as they usually carry more combined upstream traffic. The second cause is link-level performance aspects such as contention, interference, and bit-error rate. This type of congestion occurs on the link.

Congestion in WSNs has a direct impact on energy efficiency and application QoS. For example, congestion can cause buffer overflow that may lead to larger queuing delays and higher packet loss. Not only can packet loss degrade reliability and application QoS, but it can also waste the limited node energy. Congestion can also degrade link utilization. Furthermore, link-level congestion results in transmission collisions if contention-based link protocols such as Carrier Sense Multiple Access (CSMA), are used to share radio resources. Transmission collision in turn increases packet-service time and wastes energy. Therefore, congestion in WSNs must be efficiently controlled, either to avoid it or mitigate it. Typically, there are three mechanisms that can deal with this problem: *congestion detection*, *congestion notification*, and *rate adjustment*.

Congestion Detection — In TCP, congestion is observed or inferred at the end nodes based on a timeout or redundant Acknowledgments. In WSNs, proactive methods are preferred. A common mechanism would be to use queue length [4, 5], packet service time [6], or the ratio of packet service time over packet interarrival time at the intermediate nodes [7]. For WSNs using CSMA-like Medium Access Control (MAC) protocols, channel loading can be measured and used as an indication of congestion, and measurement is a means for determining congestion as in [5].

Congestion Notification — After detecting congestion, transport protocols need to propagate congestion information from the congested node to the upstream sensor nodes or the source nodes that contribute to congestion. The information can be transmitted, for example, using a single binary bit (called congestion notification (CN) bit in [2, 4, 5]), or more information such as allowable data rate, as in [6], or the congestion degree, as in [7].

The approach to disseminating congestion information can be categorized into *explicit congestion notification* and *implicit congestion notification*. The explicit congestion notification uses special control messages to notify the involved sensor nodes of congestion such as *suppression messages* as in [5]. On the other hand, the implicit congestion notification piggybacks congestion information in normal data packets. By receiving or overhearing such packets, sensor nodes can access the piggybacked information. For example, in [2] the sensor nodes that detect congestion will set a CN bit in the header of data packets to be forwarded. After receiving packets with CN bit set, the sink learns the network status, for example, congestion or no congestion.

Rate Adjustment — Upon receiving a congestion indication, a sensor node can adjust its transmission rate. If a single CN bit is used, additive increase multiplicative decrease (AIMD) schemes or its variants are usually applied, as in [2, 5]. On the other hand, if additional congestion information is available, accurate rate adjustment can be implemented, as in [6, 7].

Loss Recovery

In wireless environments, both congestion and bit error can cause packet loss, which deteriorates end-to-end reliability and QoS, and furthermore lowers energy efficiency. Other factors that result in packet loss include node failure, wrong or outdated routing information, and energy depletion. In order to overcome this problem, one can increase the source sending rate or introduce retransmission-based loss recovery. The first approach, which is also used in event-to-sink reliable transport (ESRT) [2], works well for guaranteeing event reliability for event-driven applications that require no packet reliability; however, this method is not energy efficient compared to loss recovery. The loss recovery method is more active and energy efficient, and can be implemented at both the link and transport layers. Link-layer loss recovery is hop-by-hop, while the transport layer recovery is usually done end-to-end. Here we focus on loss recovery that consists of loss detection and notification and retransmission recovery.

Loss Detection and Notification — Since packet loss can be far more common in WSNs than in wireline networks, loss detection mechanisms have to be carefully designed. A common mechanism is to include a *sequence number* in each packet header. The continuity of sequence numbers can be used to detect packet loss. Loss detection and notification can be either *end-to-end* or *hop-by-hop*. In the end-to-end approach, such as in TCP protocol, the end-points (destination or source) are responsible for loss detection and notification. In the hop-by-hop method, intermediate nodes detect and notify packet loss.

For several reasons, the end-to-end approach is not very effective for WSNs:

- The control messages that are used for end-to-end loss detection would utilize a return path consisting of several hops, and this is not energy efficient.
- Control messages travel through multiple hops and could be lost with a high probability due to either link error or congestion.
- End-to-end loss detection inevitably leads to end-to-end retransmissions for loss recovery. However, end-to-end retransmission consumes more energy than hop-by-hop retransmission.

In hop-by-hop loss detection and notification, a pair of neighboring nodes are responsible for loss detection, and can enable local retransmission that is more energy efficient, as compared to the end-to-end approach. Hop-by-hop loss detection can further be categorized as *receiver-based* or *sender-based*, depending on where packet loss is detected. In sender-based loss detection, the sender detects packet loss on either a timer-based or overhearing mechanism. In timer-based detection, a sender starts a timer each time it transmits a packet. If it does not receive an Acknowledgment from the targeted receiver before the timer expires, it infers the packet has been lost. Taking advantage of the broadcast nature of wireless channels, the sender can listen to the targeted receiver (passively and in an indirect manner so as to detect packet loss) in order to determine if the packet has been successfully forwarded.

In receiver-based loss detection, a receiver infers packet loss when it observes out-of-sequence packet arrivals. There are three ways to notify the sender: ACK (Acknowledgment), NACK (Negative ACK), and IACK (Implicit ACK). Both ACK and NACK rely on special control messages, while IACK [8] piggybacks ACK in the packet header. In IACK, if a packet is overheard being forwarded again, this implies that the packet has been successfully received and therefore acknowledged simultaneously. Therefore, IACK avoids control message overhead and is more energy efficient. However, the application of IACK depends on whether the sensor nodes have the capability to overhear the physical channel. In the case where the transmission is corrupt or the channel is not bidirectional or the sensor nodes access the physical channel using Time Division Multiple Access (TDMA)-based protocols, IACK may not be feasible.

Loss detection and notification can also pinpoint the reason for packet loss, which can be further used to improve system performance. For example, if packet loss is caused by buffer overflow, source nodes need to reduce the sending rate. However, if channel error is the cause, then it is unnecessary to reduce the sending rate in order to maintain high link utilization and throughput.

Retransmission-Based Loss Recovery — Retransmission of lost or damaged packets can be also either end-to-end or hop-by-hop. In the end-to-end approach, the source performs retransmission. In hop-by-hop retransmission, an intermediate node that intercepts loss notification searches its local buffer. If it finds a copy of the lost packet, it retransmits the packet. Otherwise it relays loss information upstream to other intermediate nodes.

If we define the node with a cached packet as a *cache point* and the node where the lost packets are detected as a *loss point*, the hop number between them can be referred to as the *retransmission distance*. The retransmission distance is an indication of retransmission efficiency in terms of energy consumed in the process of retransmission. In end-to-end retransmission (such as in TCP), the cache point is the source node. However, in hop-by-hop retransmission, the cache point could be the predecessor node of the loss point. The end-to-end retransmission has a longer retransmission distance, while the hop-by-hop approach is more energy-efficient. However, hop-by-hop requires intermediate nodes to cache packets. The end-to-end approach allows for application-dependent variable reliability levels, like that realized by ESRT. In contrast, the hop-by-hop recovery approach is preferred if 100 percent packet reliability is required, although some applications in WSNs, such as event-driven applications, may not require 100 percent reliability from sensor node. However, it needs to be noted that hop-by-hop loss recovery cannot assure message delivery in the presence of node failure.

Since end-to-end and hop-by-hop retransmissions require the caching of transmitted packets at cache points for a possible future request for retransmission, the following question would arise: How long should a cache point buffer? This is especially important if the cache point does not receive an Acknowledgment. For end-to-end retransmission, the cache duration should be close to round-trip-time (RTT). In wireless systems that use NACK-based Acknowledgments, NACK messages could be lost or corrupted on the reverse channel and the destination would be required to send NACK more than once. In this case, source nodes need to buffer a packet for a time duration which is longer than RTT. For hop-by-hop, the cache duration is only influenced by the total local packet-service time and one-hop packet transmission time.

Next we discuss several issues related to hop-by-hop retransmission in WSNs. First, when to trigger retransmis-

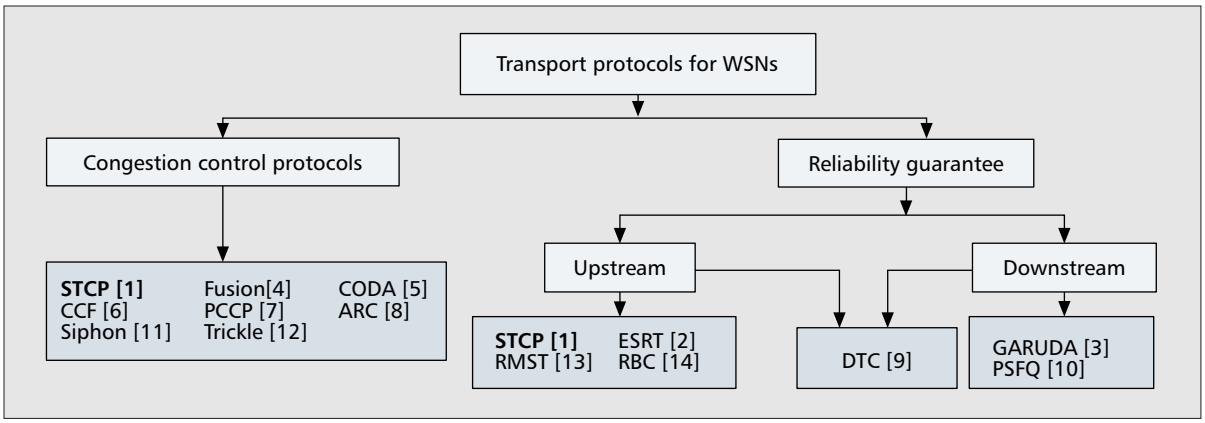

Figure 1. *Existing WSNs' transport protocols.*

sion? Retransmission can be triggered immediately upon the detection of a packet loss. This results in shorter delay, which is desirable by time-sensitive applications. However, if packet loss is caused by congestion, the immediate retransmission could aggravate the congestion situation and cause more packet losses. The second problem is related to the cache point itself. Where to cache the transmitted packets? In the hop-by-hop approach, each packet could be cached at each and every intermediate node. Given the limited memory in sensor nodes, packets may only need to be cached at selected nodes. The central issue is how to distribute cached packets among a set of nodes. The solutions, for example, in Distributed TCP Cache (DTC) [9], balance the buffer constraints and retransmission efficiency by using probability-based selection for cache points. In order to optimize retransmission efficiency, another possible approach is to cache packets at the intermediate node that is closer to the potential congested node where packet loss is more likely to arise.

Design Guidelines

In order to design an efficient transport protocol, several factors must be taken into consideration, including the topology, diversity of applications, traffic characteristics, and resource constraints. The two most significant constrains introduced by WSNs are the energy and fairness among different geographically placed sensor nodes. The transport protocol needs to provide high energy-efficiency and flexible reliability and, if necessary, QoS in terms of throughout, packet loss rate, and end-to-end delay.

Therefore, transport protocols for WSNs should have components that include congestion control and loss recovery, since these have a direct impact on energy efficiency, reliability, and application QoS. There are generally two approaches to perform this task. The first would be to design separate protocols or algorithms, respectively, for congestion control and loss recovery. Most existing protocols use this approach and address congestion control or reliable transport separately. With this separate and usually modular design, applications that need reliability can invoke only a loss recovery algorithm, or invoke a congestion control algorithm if they need to control congestion. For example, Congestion Detection and Avoidance (CODA), [5], is a congestion protocol while Pump Slowly Fetch Quickly (PSFQ) [10] provides reliable transport. The joint use of these two protocols may provide the full functionality required by the transport protocols for WSNs. In the second approach, design considerations should be taken into account to achieve a full-fledged transport protocol that provides congestion and loss control in an integrated way. For example, Sensor Transmission Control Protocol (STCP) [1] implements both congestion control and flexible reliability in a single protocol. For different applications, STCP offers different control policies to both guarantee application requirements and improve energy efficiency.

The first approach divides a problem into several subproblems and is more flexible to deal with. The second approach may optimize congestion control and loss recovery, since loss recovery and congestion control in WSNs are often correlated. For example, congestion on contention-based wireless links can lead to packet loss. The combination of CODA and PSFQ may achieve both congestion control and reliability, yet it is not well documented in the literature that such control protocols can be seamlessly integrated in an energy-efficient way. We believe there is a trade-off between an architectural/modular design (the first approach) and an integrated design with performance optimization (the second approach). The same trade-off can also be observed between the traditional protocol stack and the cross-layer optimization.

The Existing Transport Protocols for WSNS

Several transport protocols have been designed for WSNs (Fig. 1), some of which have addressed congestion or reliability only, while others have examined both. We categorize them into three types:

- Congestion control protocols
- Protocols for reliability
- Protocols considering both congestion control and reliability

Due to space constraints, only a brief summary will be provided in this article. Readers should refer to the corresponding references for further detail.

Protocols for Congestion Control

Several congestion control protocols have been proposed for upstream convergent traffic in WSNs. They differ in congestion detection, congestion notification, or rate-adjustment mechanisms (Table 1).

Among them, Fusion [4] and CODA detect congestion based on queue length at intermediate nodes, while Congestion Control and Fairness (CCF) [6] infers congestion based on packet service time. Priority-based Congestion Control Protocol (PCCP) [7] calculates a congestion degree as the ratio of packet-interarrival time and packet-service time. Siphon [11] uses the same approach as in CODA to infer congestion; in addition, this approach can detect congestion based on the perceived application fidelity at the sink. CODA uses explicit congestion notification, while others [4, 6, 7] use implicit congestion notification. In Adaptive Rate Control (ARC) [8], there is no congestion detection or notification; congestion control works as follows: an intermediate node increases its sending rate by a constant α if it overhears successful packet forwarding by its parent node. Otherwise, the intermediate node multiplies its sending rate by a factor β,

Protocols	Features		
	Congestion detection	Congestion notification	Congestion mitigation
STCP [1]	Queue length	Implicit	AIMD-like end-to-end rate adjustment
Fusion [4]	Queue length	Implicit	Stop-and-start hop-by-hop rate adjustment
CODA [5]	Queue length and channel status	Explicit	AIMD-like end-to-end rate adjustment
CCF [6]	Packet service time	Implicit	Exact hop-by-hop rate adjustment
PCCP [7]	Packet interarrival time and packet service time	Implicit	Exact hop-by-hop rate adjustment
ARC [8]	The event if the packets are successfully forwarded or not	Implicit	AIMD-like hop-by-hop rate adjustment
Siphon [11]	Queue length and application fidelity	—	Traffic redirection
Trickle [12]	—	—	Polite gossip

■ Table 1. *Congestion control protocols for WSNs.*

where $0 < \beta < 1$. ARC maintains two independent sets of α and β, respectively, for source traffic and transit traffic in order to guarantee fairness. In contrast, Fusion controls congestion in a stop-and-start nonsmooth manner. In Fusion, neighboring nodes stop forwarding packets to the congested node immediately when congestion is detected and notified. CODA adjusts the sending rate similarly to AIMD, while CCF and PCCP use an exact rate adjustment algorithm. Compared to CCF, PCCP provides priority-based fairness and overcomes the drawbacks from the use of nonwork conservative scheduling. However, there is no rate adjustment in Siphon. When congestion occurs, Siphon redirects traffic to virtual sinks (VSs) that, beside the primary low-power mote radio, have another long-rage radio used as a shortcut or "siphon" to mitigate congestion. Trickle [12] uses "polite gossip" to control traffic. In Trickle, each node tries to broadcast a summary of its data periodically. In each period, a node can "politely" suppress its own broadcasting if the number of the same metadata, which this node receives from neighboring nodes, exceeds a threshold. On the other hand, if nodes receive new code or metadata, they can shorten the broadcast period and therefore broadcast the new code sooner. In Trickle, metadata are used to describe the code that sensor nodes use, which is usually smaller in size than the code itself.

Protocols for Reliability

As shown in Table 2, some transport protocols [1, 2, 13, 14] examine upstream reliability; others, such as [3, 10], investigate downstream reliability.

In the upstream direction, ESRT discusses fidelity of the event stream and only guarantees *event reliability* through end-to-end source rate adjustment. In contrast, Reliable Multi-Segment Transport (RMST) [13] and Reliable Bursty Convergecast (RBC) [14] provide *packet reliability* through hop-by-hop loss recovery. The end-to-end source rate adjustment in ESRT follows two basic rules:

- If the current reliability perceived at the sink exceeds the desired value, ESRT will multiplicatively reduce the source rate.
- Otherwise, the source rate is additively increased if the required reliability is not met, unless there is congestion in the network.

RMST jointly uses selective NACK and timer-driven mechanism for loss detection and notification, while RBC uses a windowless block Acknowledgment with IACK. RBC proposes intranode and internode packet scheduling in order to avoid retransmission-based congestion.

In the downstream direction, traffic is multicast one-to-many. The explicit loss detection and notification meets the same problem of control message implosion as that in conventional reliable IP multicast. However, the existing approaches for reliable IP multicast do not consider several distinctive features of WSNs, especially resource constraints and application diversity. Therefore, these are not feasible for WSNs. Both GARUDA and PSFQ use NACK-based loss detection and notification, and local retransmission for loss recovery, but they design different mechanisms to provide scalability. GARUDA constructs a two-tier topology and proposes two-stage loss recovery. The two-tier topology consists of two layers, respectively, for core nodes and noncore nodes. The *hop-count* of each core node from the sink is a multiple of three. Then the first-stage loss recovery is used to guarantee that all core nodes successfully receive all packets from the sink, while the second stage is for noncore nodes to recover lost data from the core nodes. GARUDA further studies destination-related reliability. In contrast, PSFQ consists of three "operations": *pump*, *fetch*, and *report operations*. In pump operation, the sink slowly and periodically broadcasts packets to its neighbors until all data fragments have been sent out. In fetch operation, a sensor node goes into fetch mode once a sequence number gap in a file fragment is detected. It also sends a NACK in reverse path to recover the missing fragment. PSFQ does not propagate NACK messages in order to avoid message implosion. Specifically, the received NACK at an intermediate node will not be relayed unless the number of NACKs that this node has received exceeds a predefined threshold and the lost segments requested by this NACK are unavailable at this node. Finally, in report operation, the sink provides the sensor nodes' feedback information on data delivery status through a simple and scalable hop-by-hop reporting mechanism. PSFQ can be configured to use all the bandwidth and thus overcome the delay caused by the slow pump.

Protocols for Congestion Control and Reliability

STCP is a generic end-to-end upstream transport protocol. It provides both congestion control and reliability, allocating most responsibility at the sink. Intermediate nodes detect congestion based on queue length and notify the sink by setting a bit in the packet headers. This is network-assisted, end-to-end

Features	Protocols					
	End-to-end		Hop-by-hop			
	STCP [1]	ESRT [2]	RMST [13]	RBC [14]	GARUDA [3]	PSFQ [10]
Direction	Upstream	Upstream	Upstream	Upstream	Downstream	Downstream
LDN	ACK and NACK	No	NACK	IACK	NACK	NACK
LR	End-to-end	No	Hop-by-hop	Hop-by-hop	Two-tier two-stage loss recovery	Hop-by-hop
Reliability	Event and packet reliability	Event reliability	Packet reliability	Packet reliability	Packet reliability and destination-related reliability	Packet reliability

■ Table 2. *Reliable transport protocols for WSNs. LDN: loss detection and notification; LR: loss recovery.*

congestion control. One of the novelties in STCP is that it provides controlled variable reliability utilizing the diversity in applications. For example, STCP uses NACK-based end-to-end retransmission for applications producing continuous flows, and ACK-based end-to-end retransmission for event-driven applications.

Open Problems

The protocols discussed above consider either congestion control or reliability guarantees, except for STCP, which examines both. Some protocols use end-to-end and others hop-by-hop controls. Some protocols guarantee event reliability and others provide packet reliability. However, the existing protocols for WSNs have two primary limitations.

First, sensor nodes in a WSN might have different priorities since they could be installed with different kinds of sensors and deployed in different geographical locations. Therefore, sensor nodes can generate sensory data with different characteristics and have different priorities with respect to reliability and bandwidth requirements. However, most existing protocols do not consider node priority, although the recent approach in PCCP provides a priority-based congestion control. For example, most congestion control protocols guarantee simple fairness, which means that the sink needs to get the same throughput from all nodes. In addition, most reliability protocols use a single and identical loss-recovery algorithm for all nodes and applications, except for the STCP. However, the nodes and the applications may consist of diversified features and priorities, which require flexible loss recovery in order to optimize energy efficiency.

Second, the existing transport protocols for WSNs assume that single-path routing is used in the network layer. Scenarios with multipath routing are not considered. It is not clear whether they can be directly applied to WSNs with multipath routing enabled. For example, when multipath routing is utilized, with regards to a congestion control protocol, a problem would be how a sensor node should adjust its own sending rate and the sending rate of its child nodes in a fair and scalable manner. This is an issue since, with multipath routing, nodes have multiple parents and multiple path options to the sink. This problem can be even more complicated if some nodes have multiple paths while others do not.

Conclusions and Future Directions

This article has presented an overview of transport protocols and their design issues in WSNs. The ideal transport protocol for WSNs should have the following characteristics: high energy efficiency, flexible reliability, and guaranteed application-dependent QoS. Although some transport protocols have been proposed, there are several opportunities for performance optimization. In the following, we state some of those opportunities.

First, we are interested in designing WSN transport protocols that support node priority. The existing transport protocols, with the exception of STCP, consider only a single type of sensing device. It is not uncommon that a node be equipped with multiple types of sensors (e.g., temperature and humidity measurements). Thus, nodes may have different priorities and can generate sensory data with different features and requirements in terms of loss, bandwidth, and delay requirements. Different mechanisms are needed to deal with this diversity.

Second, the existing transport protocols only consider single-path routing. When multipath routing is used in the network layer, issues such as fairness arise and need to be addressed.

Third, all the existing schemes either address congestion control or loss recovery; none of them (except STCP) investigate both problems systematically. In fact, a proper congestion control should reduce packet loss and provide better throughput. Furthermore, loss recovery can enhance reliability. Therefore, transport protocols should consider both issues, together with considerations of performance optimization, energy efficiency, and other performance metrics.

Finally, the existing transport protocols rarely consider cross-layer interactions. In a WSN, link-level performance such as bit-error rate can significantly impact the performance of the transport layer protocol; similarly, routing can affect hop-by-hop retransmissions. Therefore, cross-layer optimization is highly desirable.

Acknowledgment

The research at the University of Arkansas was supported in part by AT&T Labs Research. B. Li's work was supported in part by grants from RGC under contracts HKUST6204/03E, HKUST6104/04E, and HKUST6165/05E, a grant from NSFC/RGC under the contract N_HKUST605/02, and grants from National NSF China under contracts 60429202 and 60573115.

References

[1] Y. G. Iyer, S. Gandham, and S. Venkatesan, "STCP: A Generic Transport Layer Protocol for Wireless Sensor Networks," *Proc. IEEE ICCCN 2005*, San Diego, CA, Oct. 17–19, 2005.
[2] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," *Proc. ACM Mobihoc '03*, Annapolis, MD, June 1–3, 2003.
[3] S.-J. Park *et al.*, "A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks," *Proc. ACM MobiHoc '04*, Roppongi, Japan, May 24–26, 2004.
[4] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating Congestion in Wireless Sensor Networks," *Proc. ACM Sensys '04*, Baltimore, MD, Nov. 3–5, 2004.

[5] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," *Proc. ACM Sensys '03*, Los Angeles, CA, Nov. 5–7, 2003.
[6] C.-T. Ee and R. Bajcsy, "Congestion Control and Fairness for Many-to-One Routing in Sensor Networks," *Proc. ACM Sensys '04*, Baltimore, MD, Nov. 3–5, 2004.
[7] C. Wang *et al.*, "Priority-Based Congestion Control in Wireless Sensor Networks" (to appear), *IEEE Int'l. Conf. Sensor Networks, Ubiquitous, and Trustworthy Comp.*, Taichung, Taiwan, June 5–7, 2006.
[8] A. Woo and D. C. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," *Proc. ACM Mobicom '01*, Rome, Italy, July 16–21, 2004.
[9] A. Dunkels *et al.*, "Distributed TCP Caching for Wireless Sensor Networks," *Proc. 3rd Annual Mediterranean Ad Hoc Net. Wksp.*, Bodrum, Turkey, June 27–30, 2004.
[10] C.-Y. Wan and A. T. Campbell, "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks," Proc. ACM WSNA '02, Atlanta, GA, Sept. 28, 2002.
[11] C.-Y. Wan *et al.*, "Siphon: Overload Traffic Management Using Multi-Radio Virtual Sinks in Sensor Networks," *Proc. ACM SenSys '05*, San Diego, CA, Nov. 2–4, 2005.
[12] P. Levis *et al.*, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," *Proc. 1st Symp. Networked Sys. Design and Implementation*, San Francisco, CA, Mar. 29–31.
[13] F. Stann and J. Heidemann, "RMST: Reliable Data Transport in Sensor Networks," *Proc. IEEE SNPA '03*, Anchorage, AK, May 11, 2003.
[14] H. Zhang *et al.*, "Reliable Bursty Convergecast in Wireless Sensor Networks," *Proc. ACM Mobihoc '05*, Urbana-Champain, IL, May 25–28, 2005.

Biographies

CHONGGANG WANG [M] (cgwang@uark.edu) is a post-doctoral research fellow at the University of Arkansas, Fayetteville. He received his Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications (BUPT), China. His current research interests include wireless sensor networks, wireless/mobile networks, and Internet protocols.

BO LI [SM] (bli@cs.ust.hk) is a faculty member in the Department of Computer Science, Hong Kong University of Science and Technology. He received his Ph.D. in electrical and computer engineering from the University of Massachusetts at Amherst. His current research interests are multihop wireless networks and live media streaming. He is an editor for several IEEE transactions and ACM journals, and was Co-TPC Chair for IEEE INFOCOM '04. He is currently a distinguished lecturer for IEEE Communications Society.

KAZEM SOHRABY [SM] (sohraby@uark.edu) is a professor of electrical engineering and served as head of the Department of Computer Science and Computer Engineering at the University of Arkansas, Fayetteville. He served as the director of Interdisciplinary Telecommunications Management at Stevens Institute of Technology. He is the founder of the Center for Advanced Computing and Communications Research, Networking Research Laboratory, and a principal consultant with industry. Before joining academia, he was with Bell Labs at Lucent and AT&T. He has more than 20 pending and granted patents and more than 60 publications, books, and book chapters. He currently serves as an IEEE Communications Society Director and served as its President's representative on Committee on Communications and Information Policy (CCIP). He received Ph.D., M.S., and B.S. degrees, all in electrical engineering, and an M.B.A. from the Wharton School, University of Pennsylvania.

MAHMOUD DANESHMAND (daneshmand@research.att.com) is a senior technical consultant and director of university collaborations at AT&T Labs Research, and an affiliate professor of the School of Technology Management and Computer Science at Stevens Institute of Technology. He has more than 25 years of teaching, research, and management experience in academia and industry including Bell Laboratories, AT&T Labs, University of California at Berkeley, University of Texas at Austin, Tehran University, and New York University. He has a Ph.D. and an M.A. in statistics from the University of California, Berkeley, and M.S. and B.S. degrees in mathematics from the University of Tehran.

YUEMING HU (ymhu@scau.edu.cn) is a professor in the College of Information at the South China Agricultural University, Guangzhou, China.