Context-Aware Protocol Engines for Ad Hoc Networks

J. J. Garcia-Luna-Aceves, University of California, Santa Cruz

Marc Mosko, Ignacio Solis, and Rebecca Braynard, Palo Alto Research Center

Rumi Ghosh, University of California, Santa Cruz

ABSTRACT

The protocols and architectures of the ad hoc networks of today reflect the severe memory and processing constraints that were imposed on computing equipment that was dedicated to communication tasks 40 years ago. As a result, most protocols are decoupled from the physical medium, each protocol layer operates independently of others, and processing and storage "inside" the network is kept to a minimum. We present the design and performance of a new approach to packet switching for MANETs, which we call a context-aware protocol engine. With a CAPE, nodes disseminate information in the network by means of context-aware packet switching, which enables the statistical multiplexing of bandwidth and the processing and storage of resources using integrated signaling that covers channel access, routing, and other functions, to share and store the context within which information is disseminated. Data packet headers consist of simple pointers to their context, and elections and opportunistic reservations integrated with routing are used to attain high throughput and low channel-access delay.

INTRODUCTION

The protocols and architectures used in mobile ad hoc networks (MANETs) today [1] still reflect the severe memory and processing constraints imposed on computing equipment dedicated to communication tasks 40 years ago [2]. These constraints forced the Advanced Research Projects Agency Network (ARPANET) protocols to be organized into a stack where they were decoupled from the physical medium. Additionally, protocol layers operated independently, and in-network processing and storage was kept to a minimum. This approach led to *in-band*, *in-packet* signaling where routers (packet switches) maintain the minimum amount of information required to forward packets (e.g., a next hop). Hosts attach headers that contain all of the control information for each layer in the protocol stack to data payloads. Each packet (datagram) is switched independently from other packets and also from the intent or payload type of the packet. All network content storage and processing occurs at hosts on the edge of the network.

Given the success of the Internet Protocol (IP) Internet, datagram switching, based on protocol stacks, was arguably the most sensible approach at the time the ARPANET was created. Furthermore, it is still adequate for wired segments of the Internet where link over-provisioning is feasible. However, the *stacked datagram* approach is not ideal for current MANETs with node mobility, radio channel characteristics, and a relative scarcity of bandwidth (compared to wired). Today, the in-network processing and storage power available even in small mobile nodes (e.g., personal digital assistants [PDAs], cellular phones) are orders of magnitude larger than what was available inside a network more than 40 years ago. Although there were compelling cost reasons for a division of labor between hosts and routers, today devices should not be subject to the constraints of the past. Wireless portions of the Internet must be ubiquitous and multi-functioned, acting as both network router and host to provide the content a user requests. This does not necessarily translate high network performance to that of the utilization on a given link. Interestingly, despite hardware advances, new approaches to packet switching in MANETs adhere to at least some of the original approach to packet switching introduced in the ARPANET.

In this article, we demonstrate that context-aware statistical multiplexing of network resources is far more effective (in some cases, by orders of magnitude) than implementing protocol stack-datagram switching. We introduce the context-aware protocol engine (CAPE) as an instantiation of context-aware packet switching for MANETs. The approach we advocate in CAPE is the exact opposite of current protocol architectures. A CAPE is based on nodes storing the entire packet-switching context, where packets contain a payload and a [[*pointer*]] for binding to the stored context. A CAPE employs *in-band*, *off-packet* integrated signaling, where signaling and control information (context) is sent before the corresponding data packets on the same link are sent. This context contains all of the control information required to integrate scheduling, routing, and congestion control using a single protocol. A key advantage of integrating the signaling for the operation of scheduling and routing is that shared channel-access schedules incorporate the constraints imposed by multiple-hop flows and network-level knowledge that neighbors of a node should receive the transmission. A CAPE utilizes the context-aware schedule access (CASA) protocol for signaling and channel access. CASA channel access combines distributed fair elections and a reservation mechanism. The elections determine which nodes are allowed to use or bid for available time slots but remove the requirement for direct node-to-node coordination. Reservations provide channel access-time guarantees to those nodes that successfully accessed the channel and must persist in using the same time slots.

The impact of stacked-datagram design in MANETs is most evident in the low shared-bandwidth efficiency and large overhead incurred with context-free packet switching. Accordingly, although a CAPE spans all the layers of a traditional protocol stack, this article focuses primarily on the benefits obtained with context-aware channel access and integrated signaling. We compare the performance of CASA and the improvements in overhead that are attributed to integrated signaling against the use of traditional protocol stacks based on the IEEE 802.11 distributed coordination function (DCF). The results clearly show that the use of a contention-based channel-access discipline, which in essence attempts to emulate "Ethernets in the sky," is simply not applicable to MANETs with voice and data traffic. The context-aware channel access used in a CAPE provides a solid foundation for the support of multiple media in MANETs. The overhead incurred with datagram switching, implemented with layer-independent signaling, can be substantial when either the payload per packet is small or the duration of an end-to-end flow is relatively long (involving many packets). Our results indicate that the integrated signaling in a CAPE provides reduced overhead even when the signaling associated with route maintenance and end-to-end transport is not considered.

Improving the Transition to a Wireless Medium

Considerable work in the past attempted to improve ad hoc network efficiency and scalability. We organize this work into protocol-header compaction or compression, cross-layer optimizations, and new protocol architectures.

Protocol-header compaction proposals focus on reducing the headers required for specific protocols without incurring any information loss, whereas compression proposals allow some loss of information. Header-compression or -compaction approaches are based on at least one of the following observations:

- Many header fields of packets in the packet stream of an end-to-end session are the same (e.g., source and destination address, port, version, protocol, flow label, hop limit).

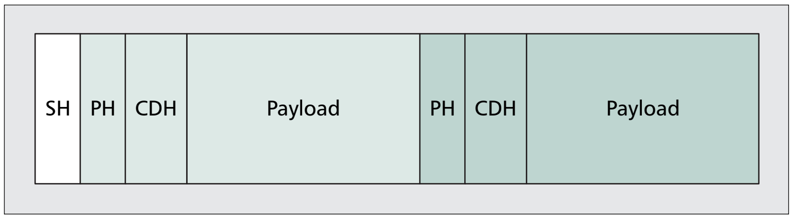


Figure 1. *CASA packet.*

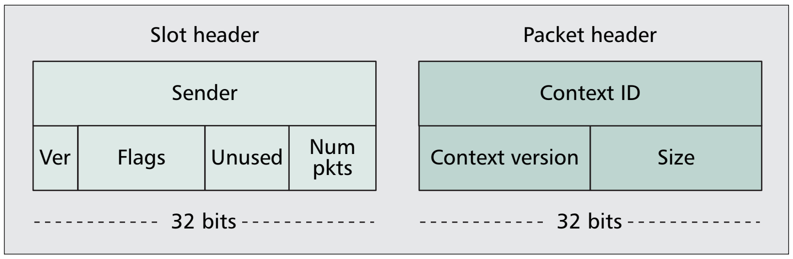


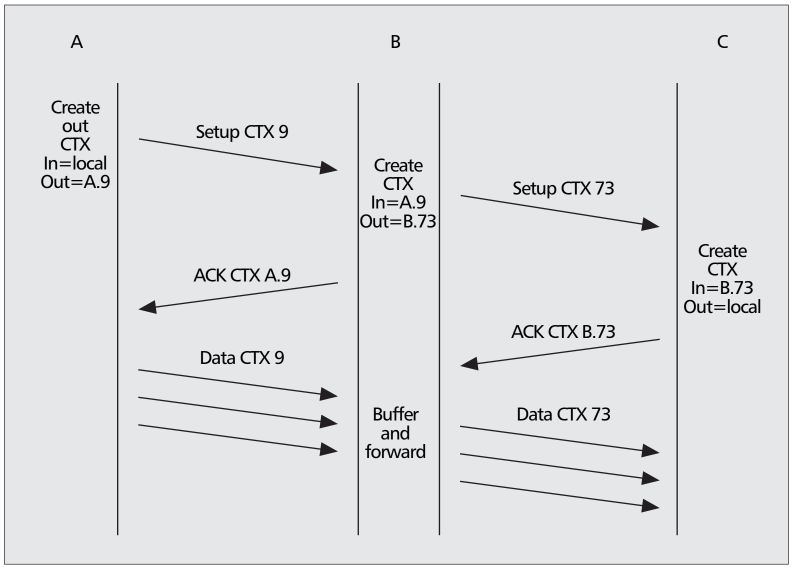
Figure 2. *CASA packet header.*

- Nodes can use local identifiers instead of globally unique identifiers, provided that they maintain a mapping for them.
- Protocol headers unnecessarily carry all the fields that the processing of a packet may require.

Most of the attention was given to the overhead of Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and IP in ad hoc networks [3, 4]. What is most striking about all the approaches to date is the continued assumption of a layered-protocol architecture based on in-band, in-packet signaling in which one layer encapsulates the higher layer, and all protocol headers are included in each packet. Simply, the goal is to attempt to reduce the overhead of specific protocols defined within that architecture. Consequently, the benefits of cross-layer interaction are not fully exploited.

Because the characteristics of the physical layer impact the performance of the entire protocol stack in an ad hoc network, recent work attempts to increase efficiency by integrating the operation of multiple layers. To date, cross-layer optimization schemes have focused on exploiting advances at the physical layer and have addressed the following: using multiple channels at the link and network layers, taking advantage of more sophisticated receivers, and utilizing relaying nodes for information processing [5, 6]. These prior results demonstrate the benefits from dynamic approaches to channel division (in time, frequency, and space) and the requirement for concurrent transmission scheduling around receivers, based on their characteristics and nearby channel state, to fully take advantage of multi-packet reception (MPR). Hence, truly scalable protocol architectures for ad hoc networks must consider the use of scheduled channel access.

Very few new architectures were proposed to improve the performance of ad hoc networks. The majority of the architectures proposed for ad hoc networks in the past focused on organiz-



■ **Figure 3.** *Context setup exchange.*

ing nodes into clusters (identified either by a cluster identifier or a node identifier) or into a connected backbone that links all nodes to reduce signaling overhead. Recently, however, Ramanathan [7] proposed an architecture based on three layers: a relay-oriented physical layer, a path-access-control (PAC) layer, and a collaborative transport layer. While Ramanathan's proposal captures many of our goals in a CAPE, it has some limitations. The PAC layer is much more subject to unfair access to resources than the current 802.11 DCF. This imbalance results from the required physical-layer resource-reservation handshake that does not incorporate any information about the context of the transmitted flows. In addition, although cooperation among nodes is important, it may not be possible for the transmitters to have accurate channel-state information with quickly moving nodes. In a CAPE, fast moving nodes could be handled by using an election contention area scaled by relative velocity, such that node IDs of fast movers are disseminated over a larger subset of the network. In tactical networks with geo-coordinates, the increased flooding could be scoped to the likely trajectory of a node.

Context-Aware Protocol Engine

In contrast to a wired network, scheduling, routing, congestion control, and retransmission control are very interrelated in a MANET. A transmission schedule, in effect, defines a "link" between a transmitter and a receiver. Establishing and using a route impacts scheduling by maintaining some links and decaying others. Last, established routes determine link congestion, and changes to these routes or the allocation of their traffic impacts congestion. Therefore, establishing the context within which resources are shared and information is exchanged must occur together with channel access, and channel access must be performed jointly with the other network control functions.

Accordingly, in a CAPE, we substitute the traditional protocol stack with:

- A context database that stores the context within which user information is disseminated and network resources are shared.
- The CASA protocol, which is used to access the shared channel and disseminate all context and user information.
- A set of network-control algorithms.

The context database information includes the flows competing for shared bandwidth, the nodes capable of causing interference around receivers, link characteristics, node positions (if available), transmission schedules and routes, other characteristics of the environment that also can help define the context, and state information used by network-control algorithms.

CASA integrates the signaling required for channel access, routing, congestion control, and retransmission control. Instead of having a medium access control (MAC) protocol, a unicast routing protocol, a multicast routing protocol, and a congestion control protocol, where each carries its own independent signaling, nodes exchange their context with one another using CASA. This single version of the context includes all the control information required to attain integrated scheduling, routing, and congestion control. Although CASA supports signaling for multiple functions, the algorithms used to implement different network control functions are *not* integrated into a single algorithm, given that such an optimization problem would be much too complex.

To populate the context database, the CAPE nodes exchange packets to build local network state. These context packets are transmitted through other CASA-like packets and do not require a special out-of-band exchange mechanism. Environment context packets contain information about local neighbors and routes. They occur periodically and maintain routing tables and the neighborhood information required by the scheduler. Flow context packets relate specific data flow information. They are in charge of set up, update, and teardown of the context information required to forward the actual data.

CASA is a time-slotted MAC layer that uses a slot header and a per-packet packet header in aggregated slots. Both header types are 64 bits. The slot header identifies who the sender is, what version of CASA is used, control flags, and the number of packets in the slot. Packet headers contain the identifier of the context within which the packet should be processed, the version of the context identifier, and the size of the packet. This information determines the format of the context-dependent header (CDH) and how to process the packet. An example of data that might be part of the context dependent header is a *link sequence number*.

To establish a data flow, first a node must transmit context set-up packets to the next hop and receive a corresponding context acknowledgment (Fig. 3). In a CAPE, a source identifies a flow with a globally unique flow ID (FID) by appending a local counter to its node ID. The FID is propagated through the network as part of connection set up. The FID takes a role akin to an IP socket descriptor: it uniquely identifies a flow between end points. After a

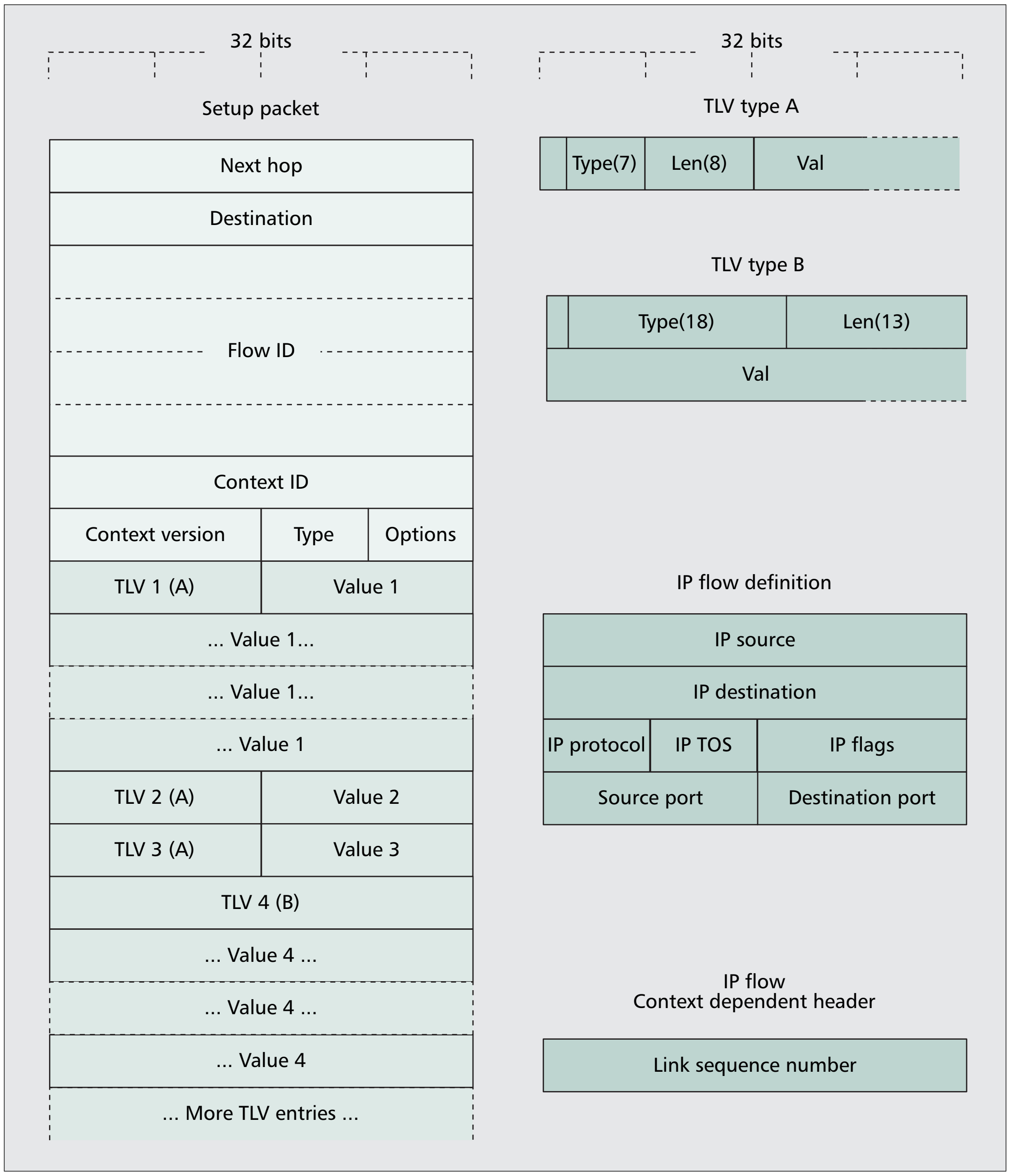


Figure 4. *CASA packets and fields.*

There are two types of TLV entries differentiated by the first bit. Type A, for most common options, uses seven bits for type and eight bits for length. Type B uses 18 bits for type and 13 bits for length to handle the extended options.

pair of neighbors along a flow path agree on the set up, the nodes switch the flow using the context ID. However, because nodes know the FID associated with specific peer connections, nodes can multiplex a flow over multiple paths. The cost of path repairs in a CAPE is minimized because nodes can reroute a flow locally using the FID.

The context id used for switching flow maps to a stored state from the set-up packet (Fig. 4), including the next hop for the packets, flow destination, FID, context version, flow type (e.g., encapsulated IP or native CAPE), and a set of type-length-value (TLV) entries used to define the flow and create context. There are two types of TLV entries differentiated by the first bit. Type A, for most common options, uses seven bits for type and eight bits for length. Type B uses 18 bits for type and 13 bits for length to handle the extended options. Most set-up packets contain a flow definition (the IP flow definition can be seen in Fig. 4).

The context defines how to interpret the context-dependent header and includes information that must be processed specifically for a given flow. In the case of an IP flow, the header includes a field called *Link Sequence Number*. This is a sequence number for the packets of this flow over the specific link in which they are sent. It is used for link retransmissions and flow control.

Efficiently Accessing the Channel through Cooperation

CASA provides access to a shared channel by means of elections and reservations based on the context information exchanged among nodes. Nodes implement a distributed election

Terrain	2500 m × 1000 m
PHY	802.11a at 12 Mb/s
802.11e	AC3 voice, AC0 TCP
Radio model	Statistical propagation, two-ray ground pathloss, constant shadowing, –111 dB propagation limit
Voice data	56 bytes CBR (17.6 kb/s codec), 30 s exp. turnaround
Bulk data	50 HTTP sessions
CASA frame	400 slots at 0.5 ms/slot (650-byte MTU)
Contention area	4 hops
Guard interval	10 μs
Reservations	300 out of 400 slots reservable
Res timeouts	1601 ms local, 2001 ms neighbor
Res rate	At most 4 new reservations/node/frame

■ **Table 1.** *Simulation parameters.*

algorithm to select which node is allowed to use the time slots that have not been reserved. There is no requirement to configure anything in the schedule other than the number of time slots used in each frame. Nodes that win slot elections can reserve the slots and use them over time to meet channel-access time guarantees. Nodes regularly broadcast their context consisting of neighborhood and reservation information to calculate schedules and propagate reservations. Nodes utilize an in-band time synchronization protocol that does not rely on external time sources, such as generalized processor sharing (GPS).

Neighbor maintenance is used to populate an N-hop neighbor table, which is the basis for the dynamic election of time slots that have not been reserved. A neighbor update consists of a neighbor node ID and the hop count (distance) to the neighbor. When sending context, all neighbors of distance less than N (for an N-hop neighborhood) are sent at once in one long array of neighbor updates, although an incremental approach is also possible. Neighbor entries time out if not refreshed.

Time-Slot Scheduling

Many of the previous dynamic scheduling protocols [8] offer high channel utilization even at high load, but still have problems with real-time data traffic because their randomized slot assignments do not provide tight enough bounds on channel-access times. Our approach maintains high utilization without sacrificing real-time data flows.

A major difference in the time-slot elections in CASA compared to previous election-based scheduling schemes is that prior work has focused on running an election for each individual time slot, whereas elections in CASA run for an entire frame. The CASA election algorithm is based on generating a pseudo-random permutation of time slots for each node. Then these permutations are compared in slot rank order to determine which node wins the right to transmit in each time slot. Although the permutations are frame-length, the algorithm can be run in amortized time over a frame. For each frame number t, we obtain a frame key $K(t)$ using a globally known hash function $K()$. The key is used as the random number seed for generating the random permutation vectors.

Each node determines the schedule of winning node IDs given the set of random permutation vectors in the matrix $P[node][slot]$ and the set of node weights for tie-breaking. After a node generates the random permutation vectors, it scans them such that the node with the minimum rank for a time slot wins the slot. The weights could be random numbers, or they could be a deterministic value based on node ID. When used with reservations, the reservation for a slot always takes precedence over the election results.

Reservations

The present reservation scheme is designed to support voice calls and, as such, is a "hold until done" strategy for keeping a reservation until it is no longer required. When a node has data to send and is below the reservation limits, it may reserve any slot for which it wins the election. CASA limits a node in the rate of slot reservations and in the maximum number of reservable slots. When a neighbor receives a slot reservation, it propagates the reservation information over the N-hop contention area so the participating nodes can use it in the distributed election. If a reservation is not refreshed, neighbors time out stale reservations using a soft-state approach.

A node may reserve a time slot if several conditions are met. The total reservations held by a node must be less than the global maximum. There is a maximum number of new reservations a node can make per frame. Nodes can keep their slot reservation based on their traffic type and the backlog at the node. To reserve a slot, a node sets the "R" flag in the slot header *flags* field. The node continues to set the "R" flag in the time slot as long as the traffic conditions are still met. A node implicitly releases its reservation after not setting the "R" flag for *Res timeout* (Table 1). It can no longer use the slot without winning it again in an election.

When a node receives a slot header with the "R" flag, it records the slot-header node identifier, the slot number, the current time, and the last-sent timestamp (initialized to 0). As part of a context-control message, every node transmits a list of the known reservations that were updated within a threshold. The reservations are ordered by their last-sent time such that the newest (initialized to 0) and least-recently sent reservations go first. Periodically, for example, on slot 0 of each frame, a node runs a maintenance routine that purges the reservation cache of any expired reservations, based on a time threshold.

A node can hear about multiple reservations for the same time slot. It should track all such reservations because they are not necessarily in conflict. However, if the node is also one of the reserving nodes, it must perform conflict resolution, such that no two nodes in a contention area hold the reservation. We use a random tie-break based on the hash of the node ID and slot number. There are no explicit reservation acknowledgments. Reservations are assumed to be valid unless the node receives a conflicting reservation and determines it lost the tie-break.

Time Synchronization

We chose the clock-sampling mutual network synchronization (CSMNS) protocol [9] for time synchronization in CASA because of its simplicity. The basis of CSMNS is the use of a proportional controller to drive the clock at each node to a common time. In each slot header, a node transmits its timestamp as a 64-bit integer. To aid convergence, we also adopt the rule that if a node has never received a timestamp before, it adjusts its offset such that the local clock equals the received timestamp.

Performance Evaluation

In this section, we present our evaluation of a CAPE and compare its performance to 802.11e. The simulations are performed for a 50-node static mesh environment, and they evaluate how a CAPE handles a mix of HTTP and voice traffic. All simulations use the optimized link state routing (OLSR) protocol in the Qualnet simulator [10] with the parameters shown in Table 1. All traffic is peer to peer between the 50 mesh nodes. The HTTP traffic uses the built-in Qualnet HTTPlib traffic generator that models page load and user think times. Each HTTP session lasts the entire simulation time between the same two nodes. Each voice call is modeled as constant bit rate (CBR) traffic with a 30-s exponential talk time in one direction; then the direction reverses for another talk spurt. Each pair of distinct CBR end points are chosen uniformly. Some nodes may participate in more than one call.

We consider the metrics of delivery, latency, and HTTP bytes delivered. We show that a CAPE has comparable voice delivery to 802.11e with a few flows, but it is over three times better with many flows. The CAPE scheduling maintains low latency that is over an order of magnitude less delay than 802.11e while delivering more elastic TCP traffic. The graphs show the average of five independent simulation runs and the 95 percent confidence interval (assumed normal).

CASA has few collisions due to being a contention-free MAC protocol, but packets still might be dropped due to fading channels, errors due to additive noise, or collisions in the infrequent case of stale context. All CASA experiments are run with an acknowledgment scheme called *TcpAcks*. This scheme only acknowledges and retransmits TCP traffic; all voice (CBR) and routing traffic is strictly best effort. TcpAck utilizes a basic acknowledgment scheme that transmits an ACK for each TCP packet. Packets are retransmitted after 50 ms if no ACK is received by the sender. Packets are retransmitted a maximum of three times.

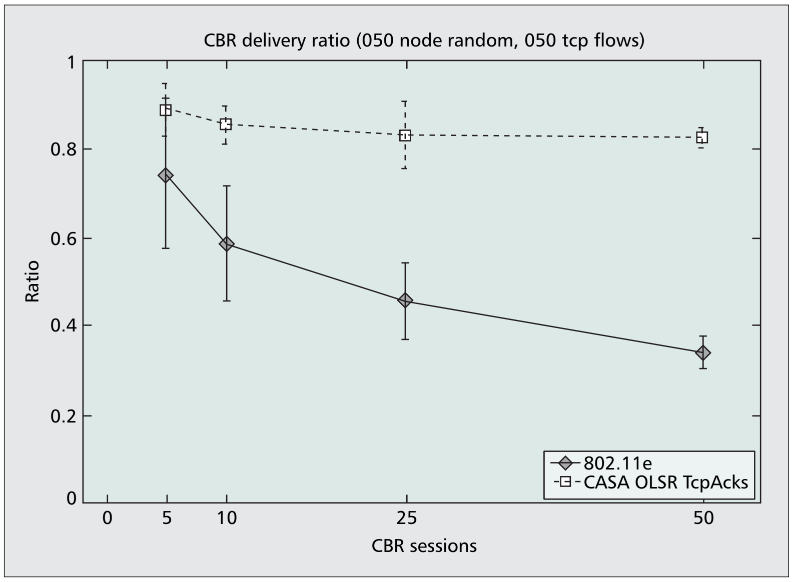


Figure 5. *Delivery ratio.*

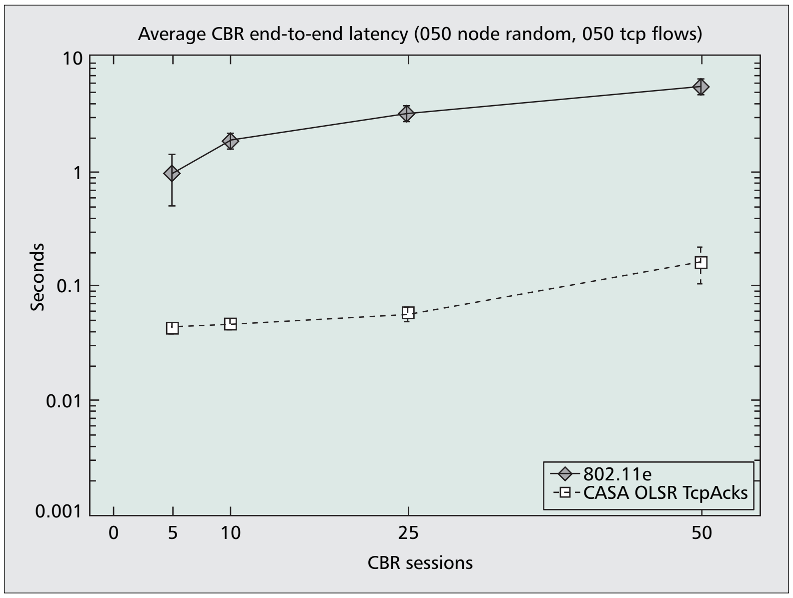
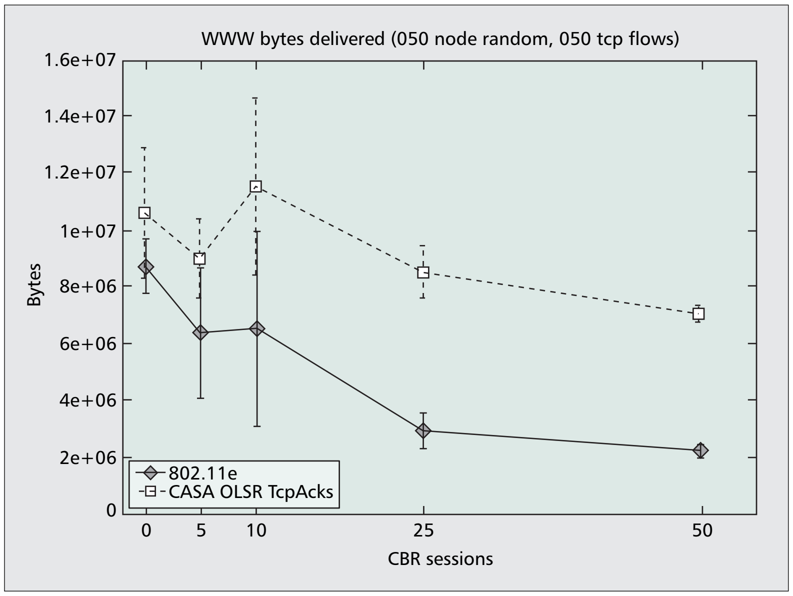


Figure 6. *CBR latency.*

Figure 5 shows the delivery ratio of CBR packets. The delivery ratio is the total number of in-order datagrams received by a destination divided by the total number of datagrams sent by all senders. The 802.11e plots use standard 802.11 RTS/CTS/ACK with retransmissions. The CAPE plot using CASA does not send ACKs or retransmit any data. Despite this lack of link automatic repeat-reQuest (ARQ), the CAPE has a fairly flat delivery ratio over all loads. CASA and 802.11 have statistically equivalent delivery ratios when there are only five CBR flows. Beyond five CBR flows, the 802.11 delivery ratio drops quickly below the CASA performance.

Figure 6 shows the end-to-end CBR latency. CASA has a significantly lower delay than



■ **Figure 7.** *HTTP bytes.*

802.11e in all cases. The 802.11e delay ranges from 1 s to 6 s. The CASA delay is between 40 msec to 60 msec for up to 25 CBR flows and then up to 160 msec for 50 CBR flows. This means that even with each node having a 16 kb/s conversation with another node and each node browsing Web pages, CASA maintains a high CBR delivery ratio and low CBR delay, whereas 802.11e breaks down after about five CBR flows.

Figure 7 shows the total HTTP bytes delivered over TCP. In these scenarios, CASA uses the TcpAck mechanism to use link ARQ on TCP packets. 802.11e and CASA deliver statistically equivalent bytes at low load (up to 10 CBR sessions), but at 25 and 50 sessions, CASA delivers about 3× the bytes of 802.11e.

CONCLUSIONS AND FUTURE WORK

We introduced the CAPE as an example of context-aware packet switching in MANETs. A CAPE is based on nodes storing the entire context within which packets are to be switched and having each data packet consisting only of its payload and a pointer to bind it to the stored context. All signaling required in CAPE is exchanged by means of a single protocol, the CASA protocol. We showed that CASA, the protocol used in a CAPE for integrated signaling and channel access, greatly improves channel utilization under heavy loads. It also performs very well in mesh traffic scenarios, enabling the multiple flows to cooperate and share the channel. To make the CAPE a reality, future work must be undertaken and includes the integration of routing with the scheduling and reservations described for channel access in CASA and the introduction of hop-by-hop congestion and retransmission control. Another important future extension of the CAPE is to incorporate fast moving nodes. Because the election scheme requires only a node ID, fast movers could inject their ID along their trajectory, ensuring correct operation.

REFERENCES

[1] B. M. Leiner, D. L. Nielson, and F. A. Tobagi, "Issues in Packet Radio Network Design," *Proc. IEEE*, vol. 75, no. 1, Jan. 1987, pp. 6–20.
[2] R. Kahn, H. Frank, and L. Kleinrock, "Computer Communication Network Design: Experience with Theory and Practice," *Networks*, vol. 2, no. 2, 1972, pp. 135–66.
[3] M. Degermark *et al.*, "Low-Loss TCP/IP Header Compression for Wireless Networks," *Proc. MobiCom '96*, 1996, pp. 1–14.
[4] I. Solis, K. Obraczka, and J. Marcos, "FLIP: A Flexible Protocol for Efficient Communication between Heterogeneous Devices," *Proc. IEEE ISCC*, July 2001, pp. 100–06.
[5] R. M. de Moraes, H. R. Sadjadpour, and J. J. Garcia-Luna-Aceves, "Many-to-Many Communication: A New Approach for Collaboration in MANETs," *IEEE INFOCOM*, May 2007, pp. 1829–37.
[6] J. J. Garcia-Luna-Aceves, H. R. Sadjadpour, and Z. Wang, "Challenges: Towards Truly Scalable Ad Hoc Networks," *Proc. MobiCom*, 2007, pp. 207–14.
[7] R. Ramanathan, "Challenges: A Radically New Architecture for Next Generation Mobile Ad Hoc Networks," *Proc. MobiCom*, 2005, pp. 132–39.
[8] L. Bao and J. J. Garcia-Luna-Aceves, "Distributed Channel Access Scheduling for Ad Hoc Networks," in *Handbook of Algorithms for Wireless Networking and Mobile Computing*, A. Boukerche, Ed., Chapman and Hall/CRC, 2006.
[9] C. H. Rentel, "Network Time Synchronization and Code-Based Scheduling for Wireless Ad Hoc Networks," Ph.D. dissertation, Carleton Univ., 2006.
[10] Scalable Network Technologies, "The Qualnet Simulator 4.0."

BIOGRAPHIES

J. J. GARCIA-LUNA-ACEVES [F] (jjgla@parc.com) holds the Jack Baskin Chair of Computer Engineering at the University of California, Santa Cruz (UCSC), and is a principal scientist at the Palo Alto Research Center (PARC). Prior to joining UCSC in 1993, he was a center director at SRI International (SRI) in Menlo Park, California. He has been a visiting professor at Sun Laboratories and a principal of protocol design at Nokia. He has published a book, more than 370 papers, and 26 U.S. patents. He has directed 26 Ph.D. theses and 22 M.S. theses since joining UCSC in 1993. He has been General Chair of ACM MobiCom 2008; General Chair of IEEE SECON 2005; Program Co-Chair of ACM MobiHoc 2002 and ACM MobiCom 2000; Chair of ACM SIG Multimedia; General Chair of ACM Multimedia '93 and ACM SIGCOMM '88; and Program Chair of IEEE MULTIMEDIA '92, ACM SIGCOMM '87, and ACM SIGCOMM '86. He has served on the IEEE Internet Technology Award Committee, the IEEE Richard W. Hamming Medal Committee, and the National Research Council Panel on Digitization and Communications Science of the Army Research Laboratory Technical Assessment Board. He has been on the editorial boards of *IEEE/ACM Transactions on Networking*, *Multimedia Systems Journal*, and *Journal of High Speed Networks*. He is listed in Marquis *Who's Who in America* and *Who's Who in the World*. He is the co-recipient of the IEEE Fred W. Ellersick 2008 MILCOM Award for best unclassified paper. He is also co-recipient of Best Paper Awards at IEEE MASS 2008, SPECTS 2007, IFIP Networking 2007, and IEEE MASS 2005, and the Best Student Paper Award of the 1998 IEEE International Conference on Systems, Man, and Cybernetics. He received the SRI International Exceptional Achievement Award in 1985 and 1989.

MARC MOSKO (mmosko@parc.com) received his Bachelor's degree in physics (1989) and a Master's degree in history (1991) from the University of California, Davis, and his doctorate in computer engineering from UCSC in 2004. He is a member of research staff at the Palo Alto Research Center (PARC). From 1992 to 2005, he worked as a network design and security consultant for Fortune 500 corporations and other companies. In 2003 while at UC Santa Cruz, he developed a routing protocol in cooperation with Nokia. His research in ad hoc wireless networks includes analysis and protocols for MAC-layer broadcasting in contention channels and on-demand routing. His MAC layer research includes several statistical and flow-based models of multihop broadcast distribution and a method for statistical optimization of multihop broadcast distribution over a contention channel. At the routing layer, he developed sev-

eral loop-free on-demand routing protocols with proofs-of-loop freedom and lockout freedom. He currently works on TDMA MAC protocols for multihop ad hoc networks. He has published 14 papers and articles, and filed over 19 U.S. patent applications. He was Local Chair for MobiCom 2008, served on the program committee for INFOCOM 2007, and was a reviewer for other conferences and journals.

IGNACIO SOLIS (isolis@parc.com) received his Ph.D. in computer engineering from UCSC. He is currently a member of research staff at PARC. His experience includes work on flexible header protocols for heterogeneous networks and prototyping architectures for protocol design. He also worked on minimizing delay for data aggregation and optimized mapping in sensor networks. With expertise in ad hoc networking, multicast routing, and collision-free wireless MAC protocols, currently he is focusing on content-centric, disruption-tolerant, and wireless-optimized network architectures.

REBECCA BRAYNARD (rbraynar@parc.com) completed her Ph.D. in computer science at Duke University, Durham, North California. She is a member of research staff at PARC. She also worked in the IBM WebSphere Technology Institute under IBM Fellow Jerry Cuomo. Her expertise is in distributed systems and networking from protocol design to experimental evaluation. Her interests include adaptive, energy-efficient, and resource-constrained protocols, overlay networks, content-based routing, and medium access control. Her thesis describes a novel approach to energy management for ad hoc sensor networks through an asynchronous, asymmetric, and adaptive duty-cycling medium access control protocol. She also collaborated with database researchers to achieve energy management goals for a sensor network through optimized application layer algorithms.

RUMI GHOSH (rumi@ieee.org) is pursuing her Ph.D. degree in computer engineering at UCSC. She received her M.S. in computer engineering from UCSC. Her current research interest is routing in wireless ad hoc networks. She worked as a member of research staff at PARC. She has 10 years of telecommunications industry experience in the areas of 3G, VoIP, and GPRS in several companies in the United States and India.