Highly-Dynamic Cross-Layered Aeronautical Network Architecture

JUSTIN P. ROHRER, Student Member, IEEE
ABDUL JABBAR, Student Member, IEEE
EGEMEN K. ÇETINKAYA, Student Member, IEEE
ERIK PERRINS, Senior Member, IEEE
JAMES P. G. STERBENZ, Senior Member, IEEE
The University of Kansas

Highly-dynamic wireless environments present unique challenges to end-to-end communication networks, caused by the time-varying connectivity of high-velocity nodes combined with the unreliability of the wireless communication channel. Such conditions are found in a variety of networks, including those used for tactical communications and aeronautical telemetry. Addressing these challenges requires the design of new protocols and mechanisms specific to this environment. We present a new domain-specific architecture and protocol suite, including cross-layer optimizations between the physical, MAC, network, and transport layers. This provides selectable reliability for multiple applications within highly mobile tactical airborne networks. Our contributions for this environment include the transmission control protocol (TCP)-friendly transport protocol, AeroTP; the IP-compatible network layer, AeroNP; and the geolocation aware routing protocol AeroRP. Through simulations we show significant performance improvement over the traditional TCP/IP/MANET protocol stack.

Manuscript received November 3, 2009; revised May 14, 2010; released for publication September 15, 2010.

IEEE Log No. T-AES/47/4/942899.

Refereeing of this contribution was handled by T. Robertazzi.

This research was supported in part by the National Science Foundation FIND (Future Internet Design) Program under Grant CNS-0626918 (PoMo).

Authors' address: Dept. of Electrical Engineering and Computer Science, The University of Kansas, 2335 Irving Hill Road, Rm. 235, Lawrence, KS 66045, E-mail: (rohrej@ittc.ku.edu).

I. INTRODUCTION AND MOTIVATION

Highly-dynamic airborne tactical networks pose unique challenges to end-to-end data transmission. The current transmission control protocol (TCP)/IP-based Internet architecture is not designed to function in this environment, however this architecture is almost exclusively used within the embedded components that make up modern tactical communications systems, as well as across the global information grid (GIG) [1]. This necessitates that any domain-specific solution designed to optimize performance in a tactical environment must at the same time maintain some compatibility with the TCP/IP stack. This paper presents the design and evaluation of a protocol suite that is optimized for the tactical environment, while maintaining edge-to-edge compatibility with the legacy Internet architecture. These protocols include: AeroTP, a TCP-friendly transport protocol introduced in [2] with multiple reliability and quality of service (QoS) modes, AeroNP, an IP-compatible network protocol (addressing and forwarding) introduced in [3], and AeroRP, a routing protocol introduced in [3] and further evaluated in [4], which exploits location information to mitigate the short contact times of high-velocity airborne nodes (ANs). This protocol suite is designed to perform well in an environment in which rapidly changing topology prevents global routing convergence, as well as those in which long-lasting stable end-to-end paths do not exist.

While these protocols are designed to perform well in a broad range of highly-dynamic scenarios, the airborne test and evaluation community in particular has recognized the need to replace an aging telemetry communication architecture with a full multihop network protocol suite such as the one described in this paper. Traditionally, telemetry communication has consisted primarily of point-to-point links from multiple sources to a single sink. More recently, with the increasing number of sources in the typical telemetry test scenario, there is a need to move to networked systems in order to meet the demands of bandwidth and connectivity. This need has been recognized by various groups, including the Integrated Network Enhanced Telemetry (iNET) program for major range and test facility bases (MRTFB) across the United States [5]. The current TCP/IP-based Internet architecture is not designed to address the needs of telemetry applications [6] and there remain a number of issues to be solved at the network and transport layers [7]. In particular, given the constraints and requirements of the aeronautical environment, the current Internet protocols are not suitable in a number of respects. These constraints include the physical network characteristics such as topology and mobility that present severe challenges to reliable end-to-end communication. In order to build a resilient [8] network infrastructure, we need cross-layer enabled protocols at the transport, network, and MAC layers that are particularly suited for airborne networks. At the same time, there is a need to be compatible with both TCP/IP-based devices located on the ANs as well as with ground-based control applications. Therefore, the new protocol suite must be fully interoperable with TCP/UDP/IP via GWs at the telemetry network edges. Due to the limited bandwidth in telemetry networks and a priori knowledge of communication needs of a given test, the iNET community is developing a TDMA (time division multiple access)-based MAC for this particular environment [9]. We revisit the telemetry-range case study later in the paper to illustrate several features of our protocol suite.

It is important to note that while tactical networks constrain some aspects of network operations, there are also aspects that can be exploited by domain-specific protocols, such as the knowledge of the AN location and trajectory. Previous research has developed several intelligent network protocols in the context of mobile ad hoc networks (MANETs) and wireless sensor networks (WSNs) that attempt to exploit additional information available [10, 11]. However, in order to achieve this, we need to facilitate cross-layering across the multiple layers. For example, location and trajectory information can be used to find better paths if there exists a mechanism, either implicit or explicit, for information exchange between the physical and network layers. As generally recognized, strict layering in the network stack is not particularly suitable for wireless networks due to mobility, limited bandwidth, low energy, and QoS requirements. Therefore, it is commonly agreed upon that a tighter, more explicit, yet careful integration amongst the layers will improve the overall wireless network performance in general; and in the case of highly-dynamic, bandwidth-constrained networks may provide the only feasible solution that meets the requirements of tactical applications.

In this paper we discuss a domain-specific suite of protocols that are designed to address the specific challenges of aeronautical telemetry. AeroTP is a transport protocol designed with several reliability modes to address the requirements of different traffic classes. This relies on the new network protocol AeroNP, which is fundamental to the architecture because it enables explicit cross-layer interactions between layers by passing congestion, QoS, and packet corruption information up and down the protocol stack. Furthermore, its header carries node and device identifiers, along with location and trajectory information that is critical for the routing protocol. Lastly, we define a location-aware, highly-adaptive routing algorithm AeroRP that utilizes the node location and trajectory to route packets through the telemetry network. Simulation results

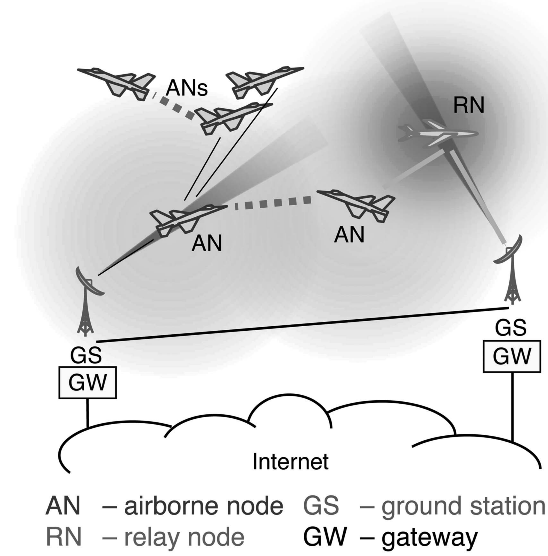


Fig. 1. Dynamic airborne tactical environment.

show that AeroRP significantly outperforms traditional MANET routing protocols that require an end-to-end path determined either proactively or on-demand. We introduce several new mechanisms to improve routing and forwarding efficiency in highly-dynamic networks. These include node discovery based on snooping, efficient directional-packet forwarding, and implicit congestion control.

The rest of the paper is organized as follows. Section II presents specific challenges to reliable network communication in the aeronautical environment through a case study of the iNET scenario. This is followed by a discussion of the current Internet architecture in Section III and its inability to meet the demands of highly-dynamic airborne networks. In Section IV we present the architecture of AeroTP and AeroNP: cross-layer aware transport and network protocols for aeronautical networks. We also present AeroRP: a lightweight routing protocol leveraging node location information. In Section V, we present simulation results comparing AeroRP with traditional MANET protocols in a highly-dynamic network. Finally, Section VI presents conclusions and directions for continuing research.

II. NETWORKING CHALLENGES IN AIRBORNE TACTICAL NETWORKS

A typical airborne tactical network as depicted in Fig. 1 consists of three types of nodes: ANs, ground stations (GSs) and relay nodes (RNs). The ANs (e.g. reconnaissance and combat aircraft, piloted or unpiloted) contain a variety of data collection devices that are primarily Internet protocol (IP) devices such as cameras, hereafter referred to as peripherals. ANs house omnidirectional antennas with relatively short transmission range. The GSs are located on the ground (stationary or portable) and typically have

TABLE I
Link Stability Analysis

Scenario	Tx Range [km]	Relative Velocity	Contact Duration [s]
Single hop best case			
AN-GS	260	206 m/s	2520
AN-AN	28	412 m/s	135
Single hop worst case			
AN-GS	185	1191 m/s Mach 3.5	300
AN-AN	18	2382 m/s Mach 7.0	15

a much higher transmission range than that of an AN through the use of large steerable antennas. In point-to-point communication mode, the GS tracks a given AN across some geographical space. However, due to the narrow beamwidth of the antenna, a GS can only track one AN at a time. The GS also houses a GW that connects the airborne network to the GIG and several terminals that may run control applications for the various devices on the AN. Furthermore, the GSs can be interconnected to do soft-handoffs from one to another while tracking an AN. The RNs are dedicated ANs to improve the connectivity of the network. These nodes have enhanced communication resources needed to forward data from multiple ANs and can be arbitrarily placed in the network. There are a number of challenges to communication protocols in this environment:

1) *Mobility*: The ANs can travel at speeds as high as Mach 3.5 (1191 m/s), possibly faster in the future; the extreme is then two ANs closing with a relative velocity of Mach 7 (2382 m/s). Because of high speeds, the network is highly dynamic with constantly changing topology.

2) *Constrained bandwidth*: Due to the limited spectrum allocated to tactical networks, and the high volume of data to be transferred, particularly for situational awareness, the network is severely bandwidth constrained.

3) *Limited transmission range*: The energy available for data transmission on some ANs is limited due to power and weight constraints, particularly with smaller vehicles, requiring multihop transmission from AN to GS.

4) *Intermittent connectivity*: Given the transmission range of the AN and high mobility, the contact duration between any two nodes may be extremely short leading to network partitioning. Furthermore, the wireless channels are subject to interference and jamming.

In Table I we use the numerical baseline values from the network characteristics of the iNET telemetry network [6] case to estimate the expected stability of the links in such a network. Even with optimistic transmission range, the contact duration between two neighboring nodes can be as low as 15 s. Note that in a multihop scenario with lower transmission power, the contact duration between an AN and GS can be even shorter. At the same time there is no maximum contact duration, so a protocol used in this environment will need to be able to make efficient use of available spectrum and manage multiple traffic priorities for long-lived flows as well as short-lived ones.

III. EXISTING PROTOCOLS AND ARCHITECTURES

Given that the tactical communications community relies in large part on existing TCP/IP-based embedded devices and communicating the data to existing IP-based applications, it is important to understand the implications of using the traditional Internet protocols (UDP, RTP, TCP, and IP) in a highly-dynamic environment. There has also been substantial research in transport protocols specific to satellite networks and routing protocols for MANETs, both of which share some characteristics with airborne tactical networks. This section considers traditional Internet protocols, as well as some domain-specific protocols and examines their suitability for this application.

A. Transmission Control and User Datagram Protocols

The most widely used transport protocol in the Internet is the TCP [12–15], which is optimized for terrestrial wired networks. TCP provides a connection-oriented reliable data-transfer service with congestion control, and uses constant end-to-end signaling to maintain consistent state at the source and destination. This introduces overhead, prevents utilizing all available bandwidth, and prevents operation in partitioned network scenarios. Each new TCP session requires a 3-way handshake before any real data is transmitted. This wastes one RTT (round-trip time) of valuable transmission time on short-lived connections such as those in the airborne network environment, and prevents the sending of any data before a stable end-to-end path exists. T/TCP (TCP for transactions) is a modification to the handshake process that bypasses the three-way handshake for subsequent connections between the same two hosts, but does not improve the process for the initial connection [16]. Even after the handshake is completed, TCP's slow-start algorithm prevents full utilization of the available bandwidth for many RTTs. This is a well-known problem in long delay scenarios such as satellite networks [17, 18], but also in highly-disconnected scenarios in which splitting application data units into many TCP segments may prevent communication. TCP also assumes that all loss is due to congestion, and its standard congestion control algorithm operates by halving the transmission rate every time there is a packet loss. This is the wrong approach for wireless networks and satellite networks [19, 20] in which noisy channel conditions are expected to be the dominant cause of packet loss [21]. TCP's flow control requires a reliable ACK stream, which limits it's ability to handle highly-asymmetric links even when the data is flowing in the high-bandwidth direction [22]. The practical limit to asymmetry for TCP flows is about 75 : 1 [23]. There is also substantial overhead with the 20 byte TCP header per packet, especially when using small segments for ACKs or to decrease the probability of suffering an errored packet. TCP was not designed with intermittent connectivity in mind; short-term link outages invoke congestion control and repeated retransmission timer back-offs, which results in an inability to detect link restoration and begin utilizing the link in a timely manner [24]. A longer link outage results in TCP dropping the connection. Varying RTT can also pose a problem for TCP, because if the actual RTT becomes much larger than the current estimate, TCP will incorrectly assume a packet loss and retransmit unnecessarily as well as reduce the congestion window. Hence, many standard TCP mechanisms are unsuitable for wireless networks in general and the dynamic airborne environment in particular.

The other commonly used Internet transport protocol is the user datagram protocol (UDP) [25]. UDP is far simpler than TCP, but does not offer any assurance or notification of correct delivery, which does not meet the reliability requirements of the tactical networks. It also does not do any connection setup, congestion control, or data retransmission and therefore does not need to maintain consistent state at both ends of the connection. UDP does not do flow control, so the need for ACKs to self-clock is eliminated completely. An extension to UDP is the real-time transport protocol (RTP) [26], which adds timing information to support real-time media but does not add any reliability or delivery assurance.

In the tactical airborne environment we expect to have multiple classes of traffic with different characteristics, different tolerance of loss, and different priorities. Neither TCP or UDP have the capability to express differentiated levels of precedence or QoS to permit the network to meet these requirements. A number of these shortcomings have been researched, and a few alternative protocols exist, such as SCPS-TP (Space Communications Protocol Standards—transport protocol) [27], from which we can draw some mechanisms but are only a partial solution.

B. SCPS-TP

SCPS-TP [27] is a set of extensions and modifications to TCP to improve operation in the space environment, particularly for satellite communications as tested in [28]. It adds mechanisms to deal with specific environmentally-induced problems, and modifies existing mechanisms to reduce undesirable behaviors. The use of the SCPS-TP options is negotiated at the time of connection establishment, which allows the SCPS-TP agent to emulate TCP when communicating with a non-SCPS peer.

In SCPS-TP the default loss assumption is a user-selectable parameter on a per-path basis, so it will not assume congestion on links in which congestion is unlikely. It also allows for signaling of congestion, corruption, and link outage both from the destination host and intermediate routers to explicitly determine the source of packet loss. SCPS-TP implements the TCP Vegas [29] slow start algorithm and congestion control based on RTT estimates. Additionally SCPS-TP queries the user for the path bandwidth-$\times$-delay product and enters congestion avoidance once the congestion window size reaches this value (similar to the congestion avoidance algorithm described in [30]). This is beneficial for paths with high RTT, however given the rapidly changing topology of an airborne telemetry network, it is practically impossible to maintain consistent RTT and bandwidth-$\times$-delay estimates. To attempt to do so would require the use of extremely conservative estimates, resulting in low utilization of the already limited bandwidth. SCPS does explicit congestion notification (ECN) using source quench SCMP (SCPS specific version of ICMP) messages [31]. It also uses an open-loop token bucket rate control mechanism [32] for each space link to avoid congestion, with the available capacity shared in the global routing structure. The highly-dynamic nature of the aeronautical environment makes it difficult to maintain globally-consistent routing information, and requires flow control to be handled locally. For loss due to corruption, SCPS-TP relies on the GS at the receiving end of each space link to maintain a moving average of the ratio of corrupted frames received and to use explicit cross-layer messages to inform the SCPS-TP destinations when that ratio exceeds a threshold. The destinations are then responsible for continuously notifying their respective sources of the corruption, during which the sources will not reduce the congestion window or back-off the retransmission timer in response to packet loss. In the case of a link outage, SCPS-TP assumes that the outage is bi-directional, so the endpoints of the space link are responsible for notifying the SCPS-TP source and destination nodes on their side of the link. SCPS-TP then enters a persist state in which it periodically probes for link restoration at which point it can resume transmission where it left off without multiple timeouts, retransmissions, or going through slow-start again.

To deal with the problem of highly-asymmetric channels, SCPS-TP reduces the number of ACKs required by TCP [33] from every other segment to only a few per RTT. This requires other TCP mechanisms such as fast retransmit [34] to be disabled. To deal with constrained bandwidth in general, SCPS-TP employs header compression and selective negative acknowledgments (SNACKs) [13, 35]. The header compression is end-to-end, as opposed to the TCP/IP header compression that is done hop-by-hop [36]. This is because hop-by-hop header compression requires a costly resynchronization process and looses all segments in flight every time a packet is lost or arrives out of order. The end-to-end compression achieves about 50% reduction in header size by summarizing information that does not change during the course of the transport session. It also avoids the problems incurred by changing connectivity because the compression takes place at the endpoints which remain constant. The SNACK option allows a single NAK [37] to identify multiple holes in the received data out-of-sequence queue. SCPS-TP also uses TCP timestamps [38] to keep track of RTTs even with lossy channel conditions, and uses the TCP window scaling option [38] so that the channel can be kept full even while recovering from losses. Many of these techniques for handling highly-asymmetric channels are applicable to the airborne telemetry network environment and are incorporated into our solution as discussed later.

While SCPS-TP solves a number of the problems associated with airborne tactical networks, and our solution uses some of the same mechanisms, we have determined that SCPS-TP is not ideal for our application because it relies too heavily on channel condition information which is either preconfigured or learned gradually over multiple end-to-end connections. This process cannot adapt adequately to the rapidly changing airborne environment, or opportunistically make use of available bandwidth on a hop-by-hop basis.

C. Internet Protocol

The traditional wired Internet uses IP at the network layer, with various routing protocols such as OSPF [39], RIP [40], and BGP [41]. TCP over IP adds a header of 40 bytes per packet. This overhead becomes significant if there are many small packets (e.g. control traffic), which is the case with the per-segment acknowledgements of TCP. The current Internet architecture is based on the fundamental assumption of long-lasting, stable links, which does not hold true for a Mach-speed airborne network which not only challenges TCP as described above, but also challenges network routing. IPs require convergence of the routes and do not natively support dynamic topologies inherent in the airborne telemetry environment. IP also does not efficiently support the inherent 2-level hierarchy caused by each AN containing a limited number of individually-addressed peripherals. Furthermore, the current architecture was not systematically designed to be a distributed solution to a global optimization problem [42] and does not support explicit cross-layer information exchange to leverage unique information available in the network such as position and trajectory.

D. Ad Hoc Routing Protocols

In order to support MANETs, several routing protocols have been developed that adapt to changes in topology. Reactive routing protocols such as AODV [43] and DSR [44] attempt to construct source-to-destination paths on demand and are not suitable because of the delay involved in finding paths and because such paths may not be valid for long enough in a highly-dynamic network. On the other hand, proactive routing protocols such as DSDV [45] and OLSR [46] forward packets on a hop-by-hop basis and depend on route convergence. This generates excessive overhead due to frequent route updates (assuming convergence is even possible) and is not suitable for a bandwidth-constrained airborne network.

There are several other protocols that adapt to mobility by forwarding packets one hop at a time without attempting to construct the entire path. These include simplistic approaches such as flooding and other greedy algorithms that send multiple copies in the network [47]. More complex routing schemes leverage specific information from the network. Most notable are the location-based routing protocols such as LAR, DREAM, SIFT, and GRID [48–52] that use GPS coordinates of the nodes to determine the next hop. LAR uses the geolocation information to limit the region in which potential routes are searched in order to reduce the overhead associated with the route discovery phase. On the other hand, DREAM uses the stored location information of the nodes to forward data in the direction of the destination. We share several mechanisms such as the location tables maintained at each node and the directional forwarding of data with the existing location-based protocols. However, previous research is not aimed at the highly-dynamic airborne networks in which the node speeds are in excess of Mach 3. The same is true of APRAM [53], which is a hybrid protocol for commercial aviation networks that utilizes geographic location to discover the shortest but complete end-to-end path between source and destination. Due to the rapidly varying connectivity in Mach-speed networks, such a mechanism is not

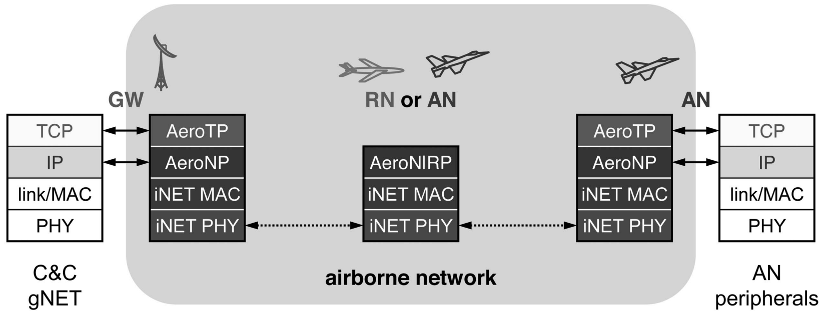


Fig. 2. Airborne network protocol architecture.

suitable. Anticipatory routing [54] tracks highly mobile endpoints that reach the reactive limit in which the speed of the nodes is comparable to time it takes for the location tracking to converge upon the position of the node. This is an extreme case that does not apply to the current scenario as shown from the contact durations in Table I. Aeronautical telemetry networks require self-organizing protocols like the one proposed in [55], but designed for high relative node mobility.

More recently, there have been new mechanisms to improve data delivery in ad hoc networks. One such mechanism is the use of network coding [56] that combines multiple packets so that they are carried together in the network, resulting in an increase of throughput and a reduction in energy cost. While the objective of network coding is to make efficient use of network resources, it requires accumulation of packets that causes increased packet delay [57]. However, this strategy is not particularly suitable for aeronautical telemetry networks in which delay becomes a significant problem due to the rapidly changing paths. Opportunistic routing [58, 59] has been proposed very recently to enhance data delivery in ad hoc networks by exploiting the promiscuous nature of wireless channels. The basic idea here is to forward a data packet to multiple neighbors who in turn collectively decide the most suitable nodes to forward the data packet. Another routing mechanism that exploits the broadcast nature of the wireless channels is the beaconless geographic routing [60] in which the geographic location of the source, destination, and neighbors is used to determine the best forwarding option. Several protocols have been proposed such as IGF [61], BOSS [62], and BLR [63] that vary in the algorithm used to select the forwarding node. However, both opportunistic routing and beaconless geographic routing, which are often implemented jointly with a MAC protocol, are more suitable for static or slowly moving nodes such that forwarding node election can be performed on a stable topology. When the node-to-node contact durations are extremely short, the performance of such mechanisms degrades significantly. We note that the routing solution proposed in this paper shares several mechanisms with the existing work on beaconless geographic routing, but is optimized for aeronautical telemetry networks.

Furthermore, airborne tactical networks require the routing protocol to be highly adaptive based on the particular mission requirements. Most existing routing mechanisms are unimodal, wherein the algorithm is optimized for a specific mode of operation. A varying set of operating conditions and service requirements justifies the need for a domain-specific multimodal protocol that inherently supports multiple modes of operation.

IV. SYSTEM ARCHITECTURE AND AERO PROTOCOLS

This section describes a new set of protocols designed for the aeronautical environment: AeroTP, a TCP-friendly[1] transport protocol; AeroNP, an IP-compatible network protocol; and the AeroRP routing protocol for highly-dynamic ANs. The major functions of each of these protocols, as well as the control-plane relationships between them are shown in Fig. 3. The communications we are concerned with can include any type of packetized information and may be directed from GS to AN, from AN to GS, or from AN to AN, and may use an intermediate RN if available. As mentioned in Section III, both the source and destination for data transmitted may be native Aero-protocol devices or TCP/IP-based systems, however the IP protocol stack is not suitable for use within the airborne network itself. To overcome this challenge without requiring a total redesign of all sensors, peripherals, applications, and workstations, we introduce the Aero gateway (AeroGW) (Acn-GW 2009) [65]. The GW concept is well established [66] as a mechanism for bridging between disparate network environments. In this case its operation is similar to TCP-Splice [67], however instead of splicing TCP with TCP, it will translate TCP (and UDP/RTP) to AeroTP and IP to AeroNP. This functionality resides in the AeroGW, which is incorporated into each GS and AN. An expected use case is shown in Fig. 2 with a GS and AN communicating using standard TCP/IP protocol

[1]Note that we use the term "TCP-friendly" in a more general sense than the established term "TCP-friendly rate control" (TFRC) [64].

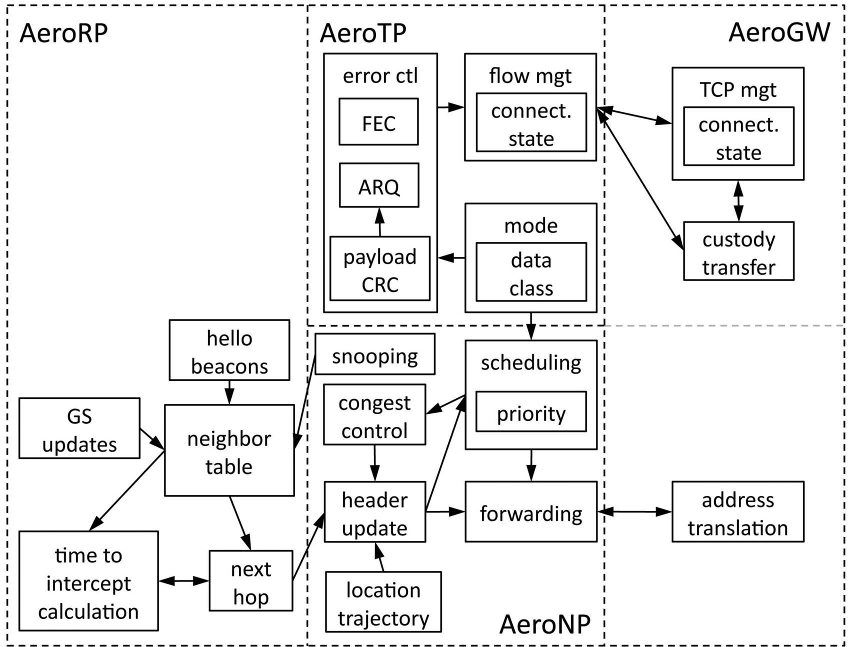


Fig. 3. Airborne network protocol functional block diagram.

stacks, which are translated to AeroTP/NP for greater dependability [68] and performance in the wireless network. It should be noted that there is no limitation preventing nodes from running the AeroTP/NP stack natively and bypassing the GW.

A. AeroTP: TCP-Friendly Transport Protocol

AeroTP is a new domain-specific transport protocol designed to meet the needs of the highly-dynamic network environment while being TCP-friendly to allow efficient splicing with conventional TCP at the AeroGWs in the GS and on the AN. Thus it transports TCP and UDP through the tactical network, but in an efficient manner that meets the needs of this environment: disruption tolerance, dynamic resource sharing, QoS support for fairness and precedence, real-time data service, and bidirectional communication. Table II identifies a number of key features of AeroTP and compares it with other modern and traditional transport protocols. AeroTP has several operational modes that support different service classes: reliable, nearly-reliable, quasi-reliable, best-effort connections, and best-effort datagrams. The first of these is fully TCP compatible, the last fully UDP compatible, and the others TCP-friendly with reliability semantics matching the needs of the mission and capabilities of the airborne network. The AeroTP header is designed to permit efficient translation between TCP/UDP and AeroTP at the GW as described in Section IVA2.

AeroTP performs end-to-end data transfer between the edges of the airborne network and either terminates at native Aero devices or splices to TCP/UDP flows at the AeroGWs. Transport-layer functions that must be performed by AeroTP include connection setup and management, transmission control, and error control, shown in Fig. 3.

1) *Connection Management and Transmission Control*: AeroTP uses connection management paradigms suited to the wireless network environment. An alternative to the overhead of the three-way handshake is an opportunistic connection establishment in which data can begin to flow with the ASYN (AeroSYN) setup message (shown in Fig. 4). The flow of data is originated by a peripheral sensor (per) as a standard TCP session, translated into an AeroTP session by the GW to traverse the airborne network, and then translated back into a standard TCP session by the GW on the ground. The TPDU (transport protocol data unit) size may be discovered using the standard path MTU discovery mechanism [69], however given the specialized nature of these networks it is expected that the best performance will be achieved by setting the peripherals to use an appropriate MTU as determined by the slot size of the underlying TDMA MAC [9]. Closed-loop window-based flow and congestion control with slow start is not appropriate to the highly-dynamic nature of this network, therefore we use an open-loop rate-based transmission control with instrumentation from the network layer and determine an initial rate, with backpressure to control congestion, as described in Section IVC for AeroNP. Error control is fully decoupled from rate control [70, 71], and is service specific as described below.

2) *Segment Structure and GW Functionality*: AeroTP is TCP-friendly, meaning it is designed to efficiently interoperate with TCP and UDP at

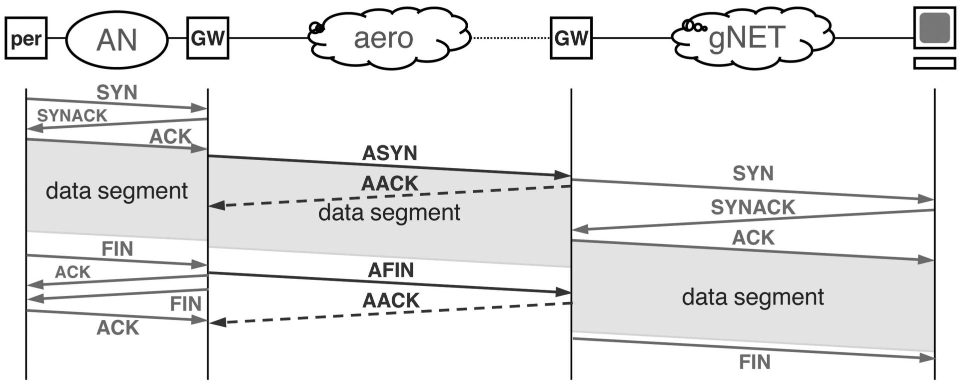


Fig. 4. AeroTP connection setup.

TABLE II
Feature Comparison of AeroTP, TP++, UDP, and TCP Variants

Feature	AeroTP	TP++	UDP	TCP NewReno	BIC/CUBIC-TCP	T/TCP	SCPS-TP
TCP Compatible	friendly	no	no	yes	yes	yes	interop
UDP Compatible	friendly	no	yes	no	no	no	no
3-way handshake	no	no	no	per-flow	per-flow	per-endpoint	per-endpoint
partial-path support	yes	no	yes	no	no	no	no
header integrity check	CRC-16	chksum	no	no	no	no	no
data integrity check	CRC-32	chksum	16-bit chksum	16-bit chksum	16-bit chksum	16-bit chksum	16-bit chksum
error correction	variable FEC	FEC	no	no	no	no	no
aggregated ACKs	yes	yes	no	optional	optional	no	yes
selective repeat	yes	yes	no	optional	optional	no	yes
negative ACKs	optional	no	no	no	no	no	optional
multipath friendly	yes	yes	no	no	no	no	no
flow control	x-layer	out-of-band signals	no	windowed	windowed	windowed	windowed
congestion ctrl	x-layer AeroNP backpressure	none	none	slow-start AIMD fast rexmit	slow-start (CU)BIC fast rexmit	estimate AIMD	estimate Vegas fast rexmit
error control	hybrid modular adaptive	hybrid modular	none	ARQ	ARQ	ARQ	ARQ
reliability modes	reliable nearly-reliable quasi-reliable best-effort	reliable quasi-reliable	best-effort	reliable	reliable	reliable	reliable

the GWs. To support this, AeroGW functionality [72, 73] provides IP-AeroNP translation [3] and TCP/UDP-AeroTP splicing. A packet may pass through two GWs on its path from source to destination. The ingress GW will convert the TCP segments to AeroTPDUs, while the egress GW will convert AeroTPDUs to TCP segments. It should be noted that ingress and egress GWs are not additional network elements in the tactical environment, but rather the GW functionality will be built into ANs and GSs. The flow diagram for the TCP to AeroTP translation that occurs at the ingress GW is presented in Fig. 5, corresponding to the flow shown in Fig. 4. The ingress GW will splice the end-to-end TCP protocol. Once the TCP SYN message is received, the GW will return a SYN ACK message. Upon receiving the SYN ACK message the source will send the TCP ACK message. The GW will transmit the AeroTP ASYN message along with the data TPDU to the destination GW after receiving the TCP ACK message. The data can piggyback on the ASYN message. The ingress GW will check the successful transmission of the data to the egress GW via incoming AeroACK (AACK) messages. If the ASYN message is delivered to the egress GW, data can continue to flow from source to destination. In the case of a failed delivery of the ASYN message, it will be sent again to preserve the end-to-end

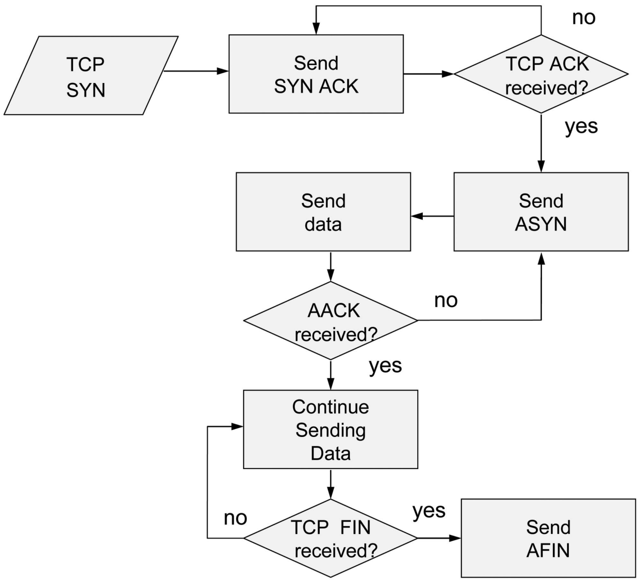


Fig. 5. TCP to AeroTP ingress GW conversion procedure.

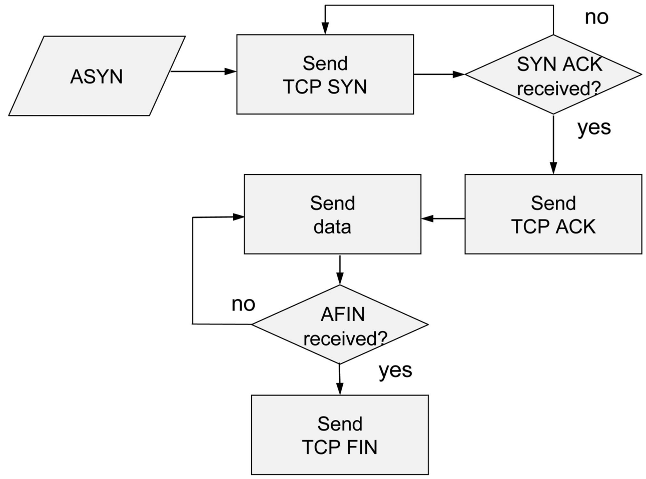


Fig. 6. AeroTP to TCP egress GW conversion procedure.

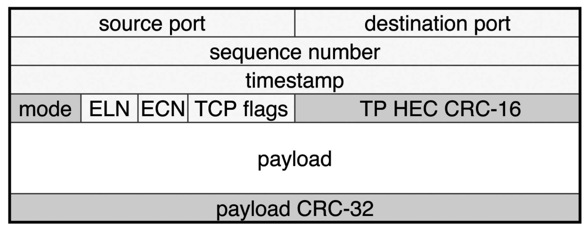


Fig. 7. AeroTP TPDU structure.

TCP semantics. Once the destination receives the application data, it will send a TCP FIN message to the GW signaling termination of the connection. The egress GW will send the corresponding AeroFIN (AFIN) message to the ingress GW to terminate the connection.

The flowchart for the AeroTP to TCP translation that occurs at the egress GW is shown in Fig. 6. The egress GW complements the splicing function by reconstructing the TCP segments. Upon receiving the ASYN message, the egress GW will send the TCP SYN message to the destination. Delivery of the TCP SYN message is checked with the SYN ACK message. If SYN ACK is not received, the egress GW will retransmit the TCP SYN message. Upon receiving the SYN ACK, the egress GW can start transmitting the data, which includes the application or control data it received from the ingress GW. Once the TCP FIN message is received from the destination, the egress GW will transmit the AFIN message to the source GW for connection termination.

The AeroTPDU is shown in Fig. 7. Since bandwidth efficiency is critical, AeroTP does not encapsulate the entire TCP/UDP and IP headers, but rather the GW converts between TCP/UDP and AeroTP headers. Some fields that are not needed for AeroTP operation but are needed for proper end-to-end semantics are passed through, such as the source and destination port number, TCP flags, and the timestamp. The sequence number allows reordering of packets due to erasure coding (as with TP++ [74]) over multiple paths or AN mobility, and is either the TCP byte-sequence number or a segment number, depending on the AeroTP transfer mode described below. The HEC (header error check) field is a strong CRC (cyclic redundancy check) on the integrity of the header to detect bit errors in the wireless channel. This allows the packet to be correctly delivered to AeroTP at the destination where a corrupted payload can be corrected on an end-to-end basis using FEC (forward error correction). A payload CRC protects the integrity of the data edge-to-edge across the airborne network in the absence of a separate AeroNP or link layer frame CRC, and enables measurement of the bit error rate (BER) for error correction code adaptation depending on the transfer mode. This method of error detection and correction implies that AeroNP does not necessarily drop corrupted packets at intermediate hops, which is a key difference from IP forwarding semantics [21, 33].

3) *Error Control and QoS-Based Transfer Modes*: Based on the application requirements, there will be a number a classes of data being transmitted over the tactical network. For this reason, AeroTP supports multiple transfer modes that are mapped to different traffic classes: reliable connection, near-reliable connection, quasi-reliable connection, unreliable connection, and unreliable datagram.

All modes except unreliable datagram are connection oriented for TCP friendliness and use sequence numbers so that packets may follow varying or multiple paths and be reordered at the AeroTP receiver.

Reliable connection mode (Fig. 8) must preserve end-to-end acknowledgement semantics from source to destination as the only way to guarantee delivery. We do this using TCP ACK passthrough, which has

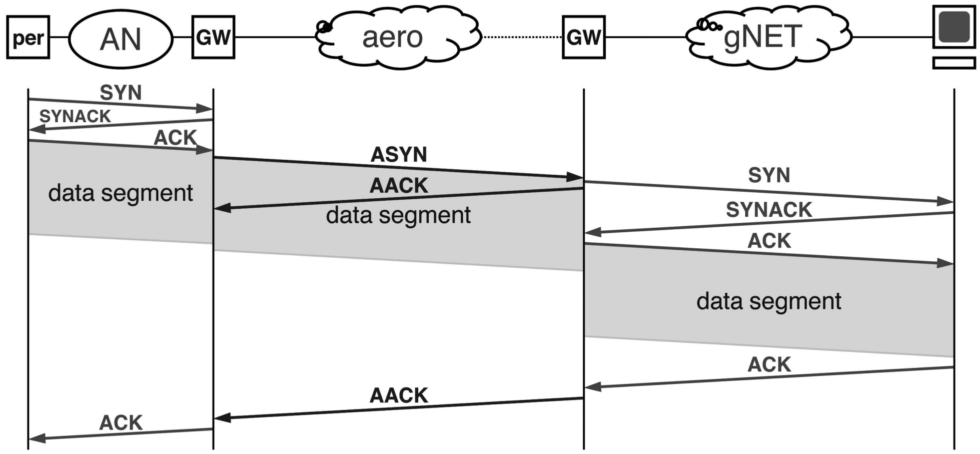


Fig. 8. AeroTP reliable connection transfer mode.

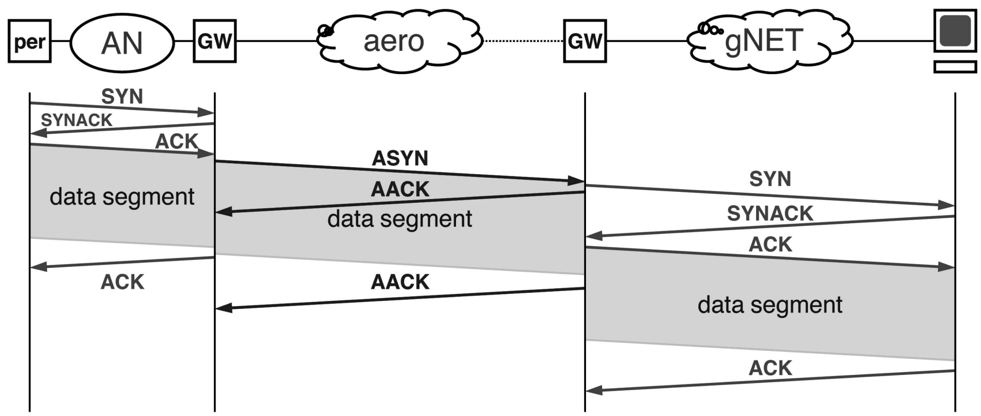


Fig. 9. AeroTP near-reliable transfer mode.

the disadvantage of imposing TCP window and ACK timing onto the AeroTP realm, but will never falsely inform the source of successful delivery.

Near-reliable connection mode (Fig. 9) uses a custody transfer mechanism similar to that used in DTNs [75, 76] to provide high reliability, but cannot guarantee delivery since the GW immediately returns TCP ACKs to the source on the assumption that AeroTPs reliable ARQ (automatic repeat request)-based delivery will succeed using SNACKs [27] supplemented by a limited number of (positive) ACKs as well as ELN (explicit loss notification) [21]. This still requires that the GW buffer segments until acknowledged across the airborne network by AeroTP, but is more bandwidth efficient than full source-destination reliability because TCP's ACK-clocked behavior only operates over the well-connected AN and ground-network (gNET) links, while allowing AeroTP to keep the assigned TDMA slots filled in the airborne network. However, the possibility exists of confirming delivery of data that the GW cannot actually deliver to its final destination.

Quasi-reliable connection mode (Fig. 10) eliminates ACKs and ARQ entirely, using only open-loop error recovery mechanisms such as erasure coding, across multiple paths if available [77]. In this mode the strength of the coding can be tuned using cross-layer optimizations based on the quality of the wireless channel being traversed, available bandwidth, and the sensitivity of the data to loss. This mode provides an arbitrary level of statistical reliability but without absolute delivery guarantees.

Unreliable connection mode (Fig. 11) relies exclusively on the link layer (FEC or ARQ) to preserve data integrity and does not use any error correction mechanism at the transport layer. Cross-layering may be used to vary the strength of the link-layer FEC.

Unreliable datagram mode (Fig. 12) is intended to transparently pass UDP traffic, and no AeroTP connection state is established at all.

B. Cross-Layer Mechanisms

Despite the fact that link-load aware routing was developed as a part of the first ARPANET routing protocol [78], cross-layered routing utilizing link and physical layer information in route selection is not widely used. The reason for this is twofold:

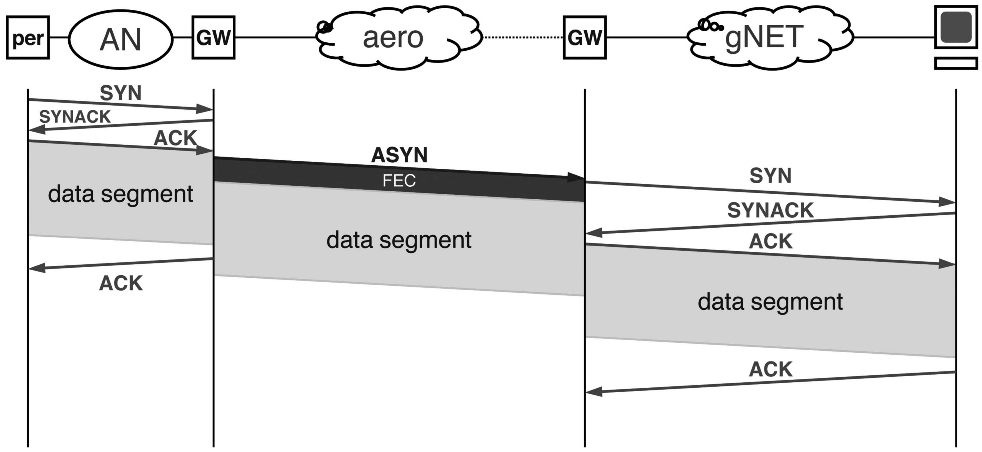


Fig. 10. AeroTP quasi-reliable transfer mode.

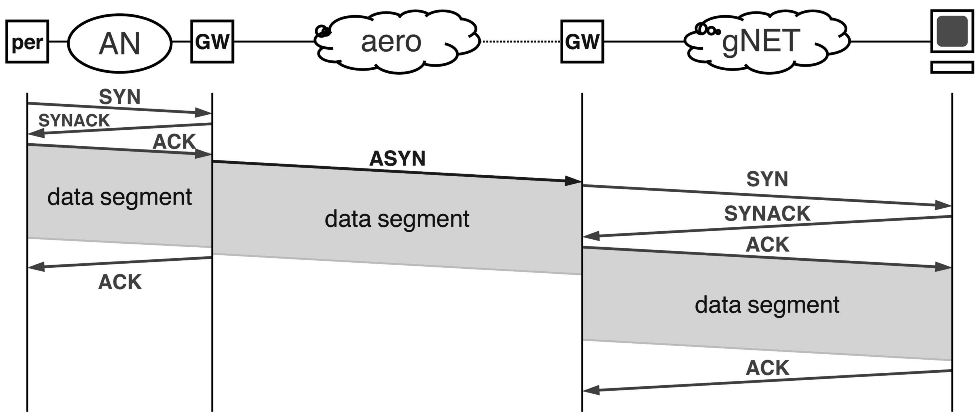


Fig. 11. AeroTP unreliable connection transfer mode.

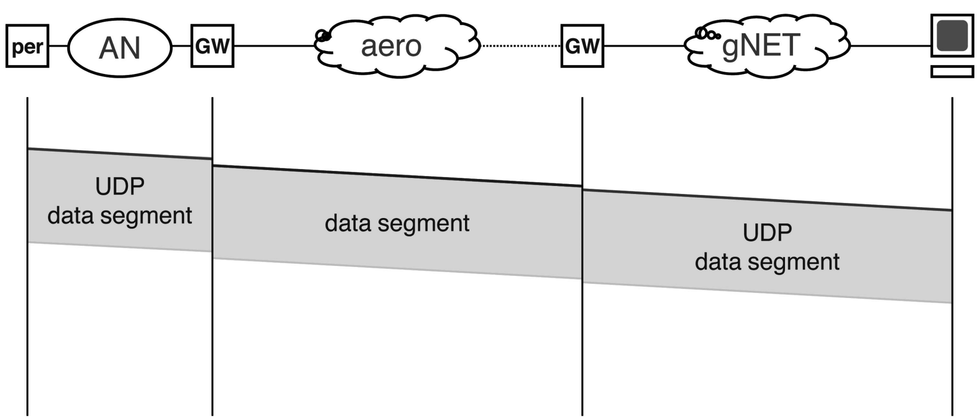


Fig. 12. AeroTP unreliable datagram transfer mode.

firstly, intelligent cross-layer aware network protocols tend to be inherently complex, and secondly in wired networks, physical links are highly reliable and are frequently overprovisioned. This has led to shortest-path being the most widely deployed routing algorithm. It has been noted that this is clearly not sufficient for effective routing in wireless networks [79], which motivates the need to exploit available information through cross-layering to make better forwarding decisions at each node.

Table III shows knobs at each layer that enable higher layers to influence certain mechanisms at lower layers, based on the information made available through dials. For example, the transport layer influences path selection through the forwarding mode knob, thus requesting a certain level of reliability for given data flow.

The airborne protocols employ cross-layer optimizations not only among the transport (AeroTP) and network (AeroNP) protocols, but also with the

TABLE III
Knobs and Dials for a Telemetry Network Stack

Layer	Knobs	Dials	Influencing Layer
mission	policy & requirements	situational awareness	command & control
application	data class	status indicators	mission
transport	reliability mode	diversity, goodput, outage	application
network	ARQ, priority	paths, loss/errors	transport
link & MAC	ARQ & FEC settings	neighbors, BER	network
physical	coding scheme	location, SNR	link

MAC and PHY layer. This involves optimizing the tradeoffs in type and strength of FEC at the PHY layer with respect to channel conditions and BER, as well as optimizing TDMA parameters and slot assignment based on the transfer mode of AeroTP and QoS parameters (precedence and service type) of AeroNP. Aditionally, the support for multicast and broadcast requires coordination of AeroNP routing with the broadcast capabilities of the MAC.

C. AeroNP: IP-Compatible Network Protocol

AeroNP is a network protocol designed specifically for the highly-dynamic airborne environment, however, given the IP-based end devices on the ground for command and control, as well as TCP/IP peripherals on the AN, it is critical for the airborne network protocol to be compatible with IP. The AeroGW converts IP packets to AeroNP packets and vice-versa. The key features of AeroNP are to provide explicit support for cross-layering messages discussed in Section IVB, reduce overhead by providing an efficient addressing mapping from IP, and provide a strong header check to decode errored payloads that could be recovered by AeroTP error-correction mechanisms.

The AeroNP packet header format, shown in Fig. 13, is 32-bits wide. The version is the AeroNP protocol version, the congestion indicator (CI) is set by each node to notify the neighboring nodes of its congestion level as discussed later. The type and priority fields specify the QoS level of a given packet. The number of QoS classes can be customized for a given scenario. Protocol is the demux protocol identifier to which AeroNP hands off the packets. In order to provide IP transparency, the ECN/DSCP (explicit congestion notification/diffserv code point) nibble is carried over from the IP header. An AeroNP packet is inserted directly into a TDMA slot, and thus contains the MAC addresses: source, destination, and next hop. Significant efficiency can be gained if the AeroNP header does not carry the 32-bit source and

<table>
<tr><td>vers</td><td>CI</td><td>C</td><td>type</td><td>priority</td><td>protocol</td><td>ECN/DSCP</td></tr>
<tr><td colspan="5">source TA MAC addr</td><td colspan="2">destination TA MAC addr</td></tr>
<tr><td colspan="5">next hop TA MAC addr</td><td>src dev ID</td><td>dest dev ID</td></tr>
<tr><td colspan="5">source TA location (opt)</td><td colspan="2">destination TA location (opt)</td></tr>
<tr><td colspan="5">length</td><td colspan="2">NP HEC CRC-16</td></tr>
<tr><td colspan="7">payload: TPDU</td></tr>
</table>

Fig. 13. AeroNP packet structure.

destination IP addresses (or the even worse 128-bit addresses for IPv6). By performing an ARP-like address resolution process, the IP address can be mapped to MAC addresses in the AeroGW. However, each AN can have multiple peripherals, each of which has an IP address. Therefore, we include a device-id field in the header, and the ⟨MAC-address, device-id⟩ tuple is mapped to the peripheral IP address at the AeroGW. While dynamic mapping procedures are possible, it is more efficient to preload the translation table at the beginning of each mission. Optionally, source and destination location are included, which can be the GPS coordinates that are used in location-aware routing. The length indicates the actual length of the header in bytes. A strong check on the integrity of the header, HEC, is included to protect against bit errors. Unlike IPs [33], the default behavior of AeroNP is to repair the corrupted bit and forward the errored packets to the transport layer instead of dropping them at the network layer. The corruption indicator (C) bit is set by AeroNP to notify AeroTP that corruption has been experienced. This permits FEC at the transport layer to correct errors in the AeroTP quasi-reliable mode, as described in Section IVA3.

D. AeroRP: Location-Aware Highly Adaptive Routing Algorithm

The small contact duration among ANs results in frequent routing changes and is indicative of the need for an intelligent multihop routing protocol, supporting reliable communication over the highly-dynamic physical topology. As discussed previously, existing routing mechanisms generate significant overhead and do not converge quickly (if ever) in the presence of frequent topology changes and hence are not suitable for highly-dynamic networks. The AeroRP routing protocol is specifically designed to address the issues related to highly-mobile aeronautical environments. We utilize a number of mechanisms that have been researched independently for use in environments with characteristics similar to those of aeronautical telemetry:

***Proactive behavior*:** AeroRP is a fundamentally proactive routing protocol, but with limited updates thereby lowering protocol overheard.

TABLE IV
Feature Comparison of AeroRP and Other Routing Protocol Categories

Feature	AeroRP	Traditional MANET (AODV, OLSR, DSDV, DSR)	Opportunistic Routing (OR, EOR)	Geographic Routing (LAR, DREAM)	Beaconless Routing (IGF, BOSS)
partial-path support	yes	no	yes	yes	yes
store & haul	yes	no	no	no	no
cross-layering	yes	no	no	yes	yes
snooping	yes	no	no	no	yes
location aware	yes	no	no	yes	yes
beaconless	optional	no	no	yes	yes
update frequency	aperiodic topology dependent	periodic or on-demand	no updates	periodic	no updates
route reconfiguration	hop-by-hop	source initiated or based on updates	hop-by-hop	based on updates	hop-by-hop
multiple op. modes	yes	no	no	no	no

***Exploits cross-layer controls*:** AeroRP is designed to exploit the explicit cross-layering support provided by AeroNP and the geographic node location and trajectory information available at nodes.

***Per-hop behavior*:** Unlike existing protocols, AeroRP forwards data per-hop based on partial local information and routes thereby avoiding the necessity for global convergence, making it especially suitable for highly-dynamic environments.

***Multi-modal*:** Military applications present a high level of variation in their operational parameters. For example, based on the security requirements of the test application, the geolocation of the nodes may or may not be available. In order to support these dynamics in operation, policies, and constraints, AeroRP provides multiple modes of operation.

1) *Protocol Operation*: The basic operation of AeroRP consists of two phases. In the first phase, each AN learns and makes a list of available neighbors at any given point in time. It utilizes a number of different mechanisms to facilitate neighbor discovery, discussed later in this section. The second phase of the algorithm is to find the appropriate next hop to forward the data packets. In order to forward the packets toward a specific destination, additional information such as location data or route updates is required. For each of these two phases the protocol defines a number of different mechanisms. The particular choice of mechanism to be used is dependent upon the mode of operation. The protocol does not specify a predefined set of discrete operational modes; the total number of supported modes is merely the combination of all the different mechanisms available. We now consider each of the two phases in more detail.

***Neighbor Discovery*:** The first objective of an AN is to determine its neighboring nodes. In order to achieve this, we use several different mechanisms with the objective to minimize overhead and increase adaptability. One or more of the following mechanisms may be used to populate the forwarding table depending upon the operational constraints.

Active snooping is the primary mechanism used by the node to locate and identify its neighbors. In the wireless TDMA network, a node that is not transmitting listens to all transmissions on the wireless channel. AeroRP adds the transmitting MAC address of each overheard packet to its neighbor table. The protocol assumes cooperative nodes and symmetric transmission ranges among ANs. This implies that if a node can hear transmissions from a node, it can also communicate with that node. Stale entries are removed from the neighbor table if no transmissions from a node are heard for a predetermined time interval related to the anticipated contact duration.

Hello beacons are used by idle nodes to advertise their presence. When neighboring nodes hear a hello beacon, they update their neighbor table appropriately. The frequency of the hello beacon is inversely proportional to the minimum calculated contact duration. For example, if the minimum contact duration is 10 s, the hello beacon is transmitted every second, however if the minimum contact duration is 100 s, the hello beacon need only be sent every 10 s.

Ground station updates may used to augment or replace active snooping in some of the mission scenarios, in which the GS has a partial or even complete mission plan. The GS sends periodic updates containing the location and trajectory vectors predicted by the mission plan to all nodes.

Security requirements may impose certain restrictions on aeronautical networks. In certain cases in which node location or trajectory is considered sensitive, individual nodes may not include this information in the header of data packets or hello updates. In this case, the GS may send location updates of all nodes on an encrypted channel. Finally, in the most secure mode, no geographic node information is available and the routes have to be discovered using traditional MANET methods, such as explicit routing updates and the exchange of node contacts between neighbors.

Given the dynamic nature of the aeronautical network, neighbor discovery not only consists of finding nodes within transmission range, but also determining the duration for which a discovered node will remain within range. Depending upon operational constraints, this information is obtained via different mechanisms: location and trajectory information is included in the AeroNP header [3], or in updates sent by the GS.

***Data Forwarding*:** After neighbor discovery, the second phase of AeroRP is for individual nodes to determine the next hop for a particular transmission. Recall that, unlike conventional protocols, AeroRP performs hop-by-hop forwarding based on partial paths without the full knowledge of the end-to-end paths [80]. Each node forwards packets such that they end up geographically closer to the destination, which will frequently be a GS in many mission scenarios.

When any given node needs to transmit data, and assuming that one or more neighbors are discovered, the data packets are forwarded to the node that is nearest to the destination as calculated from its current coordinates and trajectory. The destination location is obtained in a manner similar to that of discovering neighbors. Furthermore, in many cases the destination is the stationary GS whose coordinates are known to all ANs. The algorithm for finding the best node to forward (or handover) the data packet is given in Section IVD2

In order to avoid congestion at any given node, AeroRP utilizes the CI [81, 82] field of the AeroNP header. Each node uses the CI field to indicate its own congestion level. All packet transmissions from a node carry the CI field along with the type and priority of the data. All the neighboring nodes are thus made aware of the congestion at a given node for a given priority of the traffic and refrain from forwarding equal or lower priority traffic to the congested node.

2) *Data Forwarding Algorithm*: Let the position of ith AN, n_i be represented by the vector $\mathbf{P}_i = (x_i, y_i, z_i)$ and the trajectory is defined by the vector $\mathbf{T}_i = (s_i, \theta_i, \phi_i)$, where x, y, and z are the absolute node coordinates, $\mathbf{T}$ is the spherical direction vector (speed, inclination, and azimuth). Since the network is highly dynamic, both the position and trajectory of nodes are time dependent. For a given source-destination pair, at a given time t, let the source node n_s have the position $\mathbf{P}^t_\mathrm{s} = (x^t_\mathrm{s}, y^t_\mathrm{s}, z^t_\mathrm{s})$ and the trajectory $\mathbf{T}^t_\mathrm{s} = (s^t_\mathrm{s}, \theta^t_\mathrm{s}, \phi^t_\mathrm{s})$. Similarly the destination node n_d is represented by $\mathbf{P}^t_\mathrm{d} = (x^t_\mathrm{d}, y^t_\mathrm{d}, z^t_\mathrm{d})$ and $\mathbf{T}^t_\mathrm{d} = (s^t_\mathrm{d}, \theta^t_\mathrm{d}, \phi^t_\mathrm{d})$. If the destination happens to be a stationary GS, then $\mathbf{P}^t_\mathrm{d} = \mathbf{P}_\mathrm{d}$, $\forall t$. Finally, let the congestion status of the node be given by the vector $\mathbf{C}_i = \{\mathrm{CI}, \mathrm{priority}\}$, where CI and priority are the congestion indicator and priority fields that are extracted from the AeroNP header.

Step 1 Each node maintains two tables: a neighbor table that stores the information about the nodes that are currently in the transmission range, and a destination data table that stores the information of all destinations, which may or may not be currently in the transmission range. Initially, let the number of neighbors represented by the neighbor list N be zero, i.e., $N = \emptyset$.

Step 2 When the node receives any packet, it updates the neighbor and destination data tables. If the captured packet is an overheard transmission or hello advertisement from node n_i, the node i is assumed to be in the transmission range of the current node. Hence the neighbor list is updated as $N = N \cup \{n_i\}$. Furthermore, the macID, position, trajectory, and congestion status of the node are derived from its header and stored in the neighbor table as the tuple $\{\mathrm{macID}_i, \mathbf{P}^t_i, \mathbf{T}^t_i, \mathbf{C}^t_i\}$. If the received transmission is a GS update, each entry in the update is stored in the destination data table as the tuple $\{\mathrm{time}, \mathrm{macID}_i, \mathbf{P}^t_i, \mathbf{T}^t_i, \mathbf{C}^t_i\}$. Since the GS update may contain information on node positions in future, the entries in the destination data table are time stamped. Lastly, when a GS update is received, the location and trajectory fields of neighbor table entries are updated with the latest values.

Step 3 At the completion of step 2, assume that a given node n_0 has k discovered neighbors. From this set of k neighbors, all congested neighbors (CI bit set for priority equal or greater than the priority of the data to be sent) are removed. Furthermore, each node adds itself as the first neighbor in the list: $N_0 = \{n_0, n_1, \ldots, n_i, \ldots\}$. Assume that node n_0 wants to send a data packet to the GS n_d with position $\mathbf{P}_\mathrm{d}$. Assume that the transmission range of all nodes is R. Next, we calculate the time to intercept (TTI) for all neighbors. The TTI_i represents the time it will take for node n_i to get reach within the transmission range of the destination if it continues on its current trajectory. TTI is calculated as

$$\mathrm{TTI}_i = \frac{|\mathbf{P}^t_\mathrm{d} - \mathbf{P}^t_i| - R}{s_\mathrm{d}} \tag{1}$$

where $|\mathbf{P}^t_\mathrm{d} - \mathbf{P}^t_i|$ gives the Euclidian distance between the current location of node n_i and the destination node n_d and s_d is the component of the actual speed s_i of node n_i in the direction of the destination and is calculated as

$$s_\mathrm{d} = s_i \times \cos(\theta_i - \theta_\mathrm{d}) \tag{2}$$

where θ_d is the angle of the destination with respect to the current node position.

Step 4 Finally, the data is forwarded to the jth node, n_j such that:

$$\mathrm{TTI}_j = \min\{\mathrm{TTI}_i\} \qquad \forall i : n_i \in N_0. \tag{3}$$

The process is repeated at every node, until the data reaches the destination.

Ground stations are special nodes in this network. They listen to all the transmissions and forward packets that are destined to other GSs. In other words, GSs are universal sinks and may share the same MAC address. For uplink data, a GS forwards data to the node that is closest to the destination node. The GS is aware of the location of all nodes either from mission planning or by learning it during the test from header information in received packets.

RNs (relay nodes), if present, are always the default next-hop. They accept packets from all the ANs and forward them directly to the GS or another AN. Since the GS has narrow beamwidth and can only track one AN at a time, it is more efficient for the GS to track RNs and have individual ANs forward their data via RNs. Given the varied service requirements of tactical missions, AeroRP supports multiple modes for both open and secure scenarios.

3) *Mission Based Quality of Service*: The wireless links in the telemetry network are bandwidth constrained and may be underprovisioned for the traffic generated at any give time. Hence, it is essential to implement a QoS mechanism in this network to ensure that high priority data, such as command and control, can be reliably delivered. The AeroNP protocol uses two fields in the header to specify the QoS of packets in the network: data type (e.g. command and control, telemetry) and priority within a given type. The mission and application requirements determine the type and priority for a given data flow, which are passed to AeroNP through AeroTP via out-of-band signaling. The scheduling algorithm at nodes is weighted fair queuing based on type and priority.

4) *Broadcast and Multicast*: The AeroNP protocol supports both broadcast and multicast natively. The typical all-ones MAC address is used as the broadcast address. Similarly, a range of MAC addresses are assigned to subgroups in the network. These multicast address groups are generally preprogrammed in the nodes and GS. Note however, that given the highly dynamic nature of the network, multicast may not achieve any significant benefit over a simple broadcast in terms of efficiency for sparse networks.

5) *Congestion Control*: In a heavily loaded network multihop routing can induce severe congestion at nodes involved in multihop forwarding as well as transmitting their own telemetry data. To overcome this, AeroNP uses a simple congestion control mechanism at the network layer using CIs and back pressure. We choose these algorithms for their simplicity in their operation based on little feedback. The objective is to avoid local congestion and it does not guarantee global optimization or fairness. A more rigorous rate control mechanism such as the one proposed in [83] is not suitable here due to the highly dynamic nature of the network, in which an optimal solution would become stale by the time it is achieved.

In the first mechanism, the node uses the CI [81, 82] field to indicate its own congestion level. Even though 2 bits are assigned to the CI field, only two of the four possible values are currently used. Hence CI is toggled between `0x0` and `0x3`. All packet transmissions from a node carry the CI field along with the type and priority of the data. When the transmit queue of a node exceeds a predetermined threshold, the node sets its CI field to `0x3`. Neighboring nodes eavesdrop on the transmission and are made aware of the congestion at a given node. If a node is congested, the neighbors back off if the data that they have is of equal or lesser priority, however higher priority data is still forwarded to a congested node because the priority queue at that node will service this traffic first.

The second mechanism through which congestion control is achieved in the telemetry network is back pressure [84, 85]. As a source sends packets to an intermediate node, it simultaneously eavesdrops on that node to see if the packets are being forwarded at the same rate they are being sent. If not, and other packets are being forwarded instead, then the source can infer that the next hop it has chosen is queuing its packets due to congestion. The source node then backs off and if possible chooses an alternate next-hop. Similarly, in a multihop scenario, if a bottleneck is encountered, each intermediate hop either stops or slows down its transmissions on the congested path successively until the source of the traffic is reached.

V. SIMULATION RESULTS

This section presents results from simulations of the AeroTP and AeroRP protocols performed using ns-3 and ns-2, respectively. The performance of AeroTP is compared with TCP, and the performance of AeroRP is compared with that of DSDV and AODV.

A. AeroTP Connection Establishment

As mentioned previously, one of the drawbacks of TCP for highly-dynamic airborne environments is the three-way-handshake used for connection establishment. For this reason AeroTP is designed to establish a connection when the first data TPDU (with ASYN bit set) in a flow is received. If the first packet is lost, the connection can still be established using header information from the second or subsequent data packet, and the first packet can be retransmitted later if required by the specified reliability mode. To illustrate the difference between these two approaches, we have done simulations comparing the time required to establish a standard TCP connection, compared with an AeroTP connection.

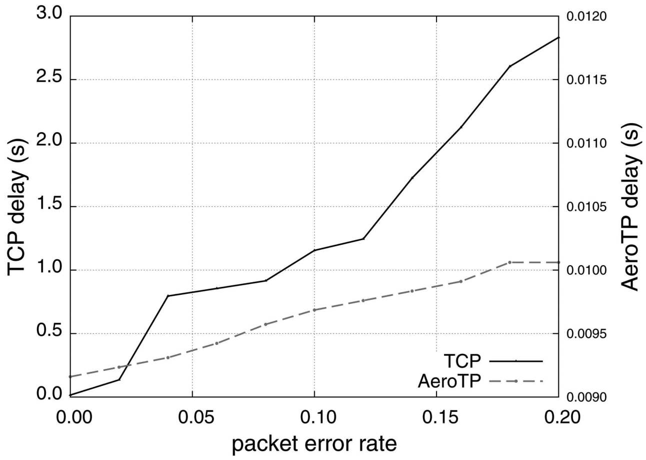


Fig. 14. TCP and AeroTP connection establishment delay.

The simulations are implemented in the ns-3 open-source simulator [86]. Each simulation consists of two nodes connected by a 10 Mbit/s link with 5 ms latency and a fixed probability of packet loss, which is varied between 0 and 20% as seen on the x-axis. Node 0 is configured as a traffic generator (TCP or AeroTP as appropriate) and node 1 is configured as a traffic sink. For each packet-loss probability point plotted, the simulations were run 100 times and the results averaged. Each simulation consists of a single connection attempt by either TCP or AeroTP. We record the delay starting when the `connection_setup` command is issued to the transport protocol, and stopping when the first data packet is received by the data sink.

Fig. 14 shows the results of these simulations. Both the TCP and AeroTP results are presented in a single plot, however, note that they are plotted against two different y-axes: TCP on the left, and AeroTP on the right. The TCP delay starts at about 20 ms when no losses occur, and increases linearly until it approaches 3 s when the packet-loss rate is 20%. AeroTP on the other hand, has a delay of 9.2 ms when no losses occur, and increases linearly to 10.1 ms when the packet-loss rate is 20%. This shows an improvement of two orders of magnitude, which will play a large role in enabling AeroTP to successfully send data over paths which only exist for a few seconds, while TCP would still be trying to establish the connection.

B. AeroRP Performance Comparison

AeroRP is implemented and simulated in the ns-2 simulator [87] and compared with performance to the traditional MANET routing protocols AODV (ad-hoc on-demand distance vector) and DSDV (destination-sequenced distance vector).[2] AODV only finds routes as needed, while DSDV updates its routing tables as the topology changes. For our routing simulation case, we revisit the telemetry test network scenario. All the ANs are assumed to be transmitting telemetry data to the GS exclusively, and to be operating within a fixed test range area served by a single GS.

1) *Topology Setup*: 60 ANs are randomly distributed over a 150 km × 150 km test range, and a single stationary sink node is located in the center of the simulation area representing an RN that is constantly tracked by a GS. The 60 ANs follow a modified random-waypoint mobility model [89] for a total of 2000 s. The pause times are zero to more accurately represent the movement patterns of aircraft. Two different test cases are simulated. In the first case each node's speed is randomly selected to be between Mach 0.3 and Mach 3.5 (100 to 1200 m/s) for each leg of the random-waypoint movement; in the second case the nodes always move at Mach 3.5. Each node has an omnidirectional antenna with a maximum range of 27.8 km (15 nmi). This yields a total coverage ratio of 6.5 : 1. The velocities and radio transmission ranges are based on the iNET architecture [6], and the node density is such that the telemetry network is not partitioned most of the time. In the simulations ANs are partitioned from the sink node an average of 6.6% of the time. In other words, the sink node and the source nodes are not in the same partition for a very small duration of time, which indicates fairly good network connectivity. The objective is to isolate the effect of routing from network connectivity.

2) *Traffic Setup*: The traffic model is such that telemetry data originating at all node is destined to a single destination (sink node) that represents the GS. The telemetry data from node is a constant data flow at a rate of 0.2 Mbit/s with 1000-byte packets resulting in 25 packets being sent per node per second, and a combined total of 1,350,000 packets for all nodes over the course of the simulation. The wireless link bandwidth is set to 11 Mbit/s so that congestion is not a factor in the results. Data transmission does not start until the 1050th second to allow thorough mixing of the nodes as well as route table population for DSDV. Data transmission stops at the 1950th second, and the simulation runs for an additional 50 s to allow buffered packets to be delivered.

3) *Performance Results*: In the first case using AODV, only 3.25×10^5 out of 1.35×10^6 packets (24%) are received by the sink node. DSDV performs better with 6.74×10^5 packets (50%) being received. With AeroRP geolocation-assisted predictive routing, 1.31×10^6 packets (97%) are received at the sink node. Fig. 15 shows the packet delivery rate (for an aggregate source rate of 1500 packets per second), and Fig. 17 shows the number of packets successfully

[2]We are in the process of converting all of our simulation to ns-3, however the MANET models are still under development by the ns-3 community [88].

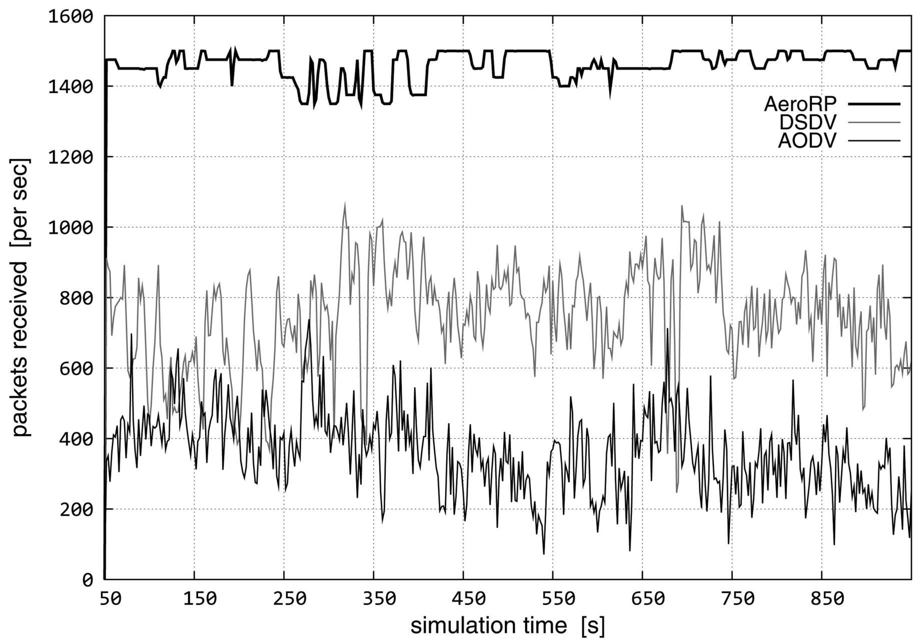


Fig. 15. Packet delivery rate for Mach 0.3 to 3.5.

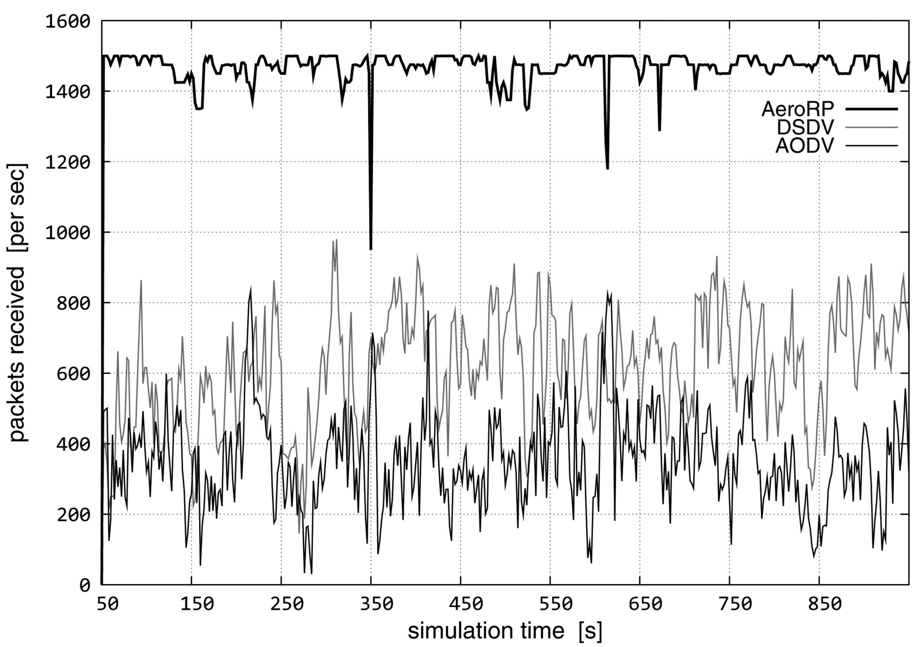


Fig. 16. Packet delivery rate for Mach 3.5.

delivered over the course of the simulation for these three protocols when the speed is varied between Mach 0.3 and Mach 3.5. Figs. 16 and 18 show the results for the second case in which nodes move at a constant speed of Mach 3.5. In this case AODV received 3.17×10^5 packets (23%), DSDV received 5.54×10^5 packets (41%), and AeroRP received 1.32×10^6 packets (97%). The packet delivery metric in Fig. 15 and 16 shows that AeroRP not only has better average performance but also reduces fluctuations in the throughput. We observe that when the convergence of the protocol does not keep up with the path fluctuations, suboptimal routes are selected leading to higher packet loss. Secondly, the selected paths remain valid for a very short duration leading to a short bursts of packet deliveries. Hence we observe noise in the the packet delivery plots for AODV and DSDV. The paths selected by AeroRP based on node location and trajectory do not experience these issues to the same extent.

The overhead incurred by AODV and DSDV for both cases is plotted in Figs. 19 and 20 in terms of aggregate bytes transmitted per second. In all cases only the signaling overhead due to routing updates

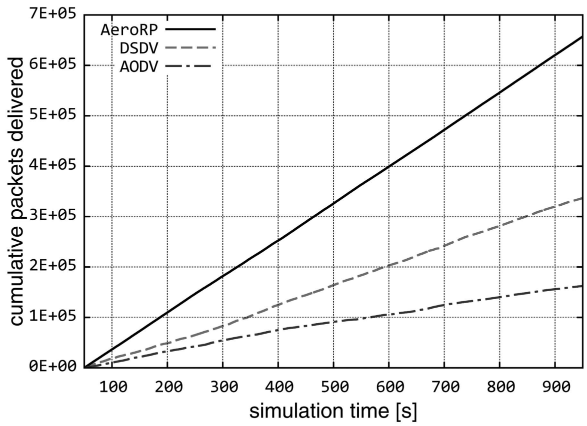


Fig. 17. Cumulative packets delivered for Mach 0.3 to 3.5.

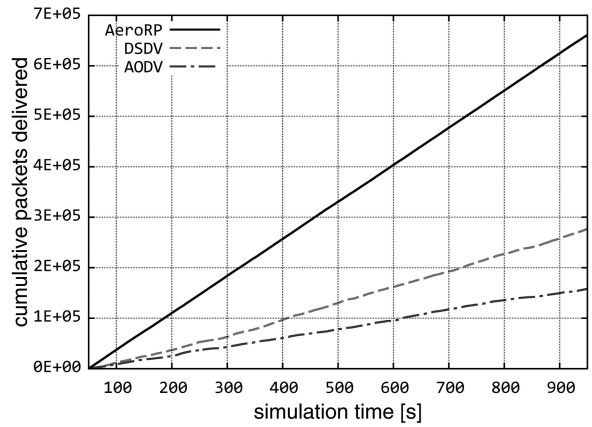


Fig. 18. Cumulative packets delivered for Mach 3.5.

is considered. AODV incurs greater overhead due to the fact that it is on-demand and therefore its overhead is directly proportional to the mobility. DSDV incurs less overhead than AODV because of the periodic nature of its update messages, which are not mobility dependent. However, as the simulations progress DSDV is unable to converge due to the highly- dynamic topology and generates increasing numbers of update messages. The overhead incurred by AeroRP varies depending on the type of route updates used to populate the routing table. These simulations focus on the snooping (with hello beacons) and GS broadcast mechanisms described in Section IVD. Snooping alone does not cause any overhead, because the traffic model is heavy enough that no `hello` messages are required. Figs. 19 and 20 show the overhead induced by using the GS to broadcast the current and predicted link-state table to all the nodes at 10 s intervals. Thus, the overhead in this case is simply the size and frequency of the GS updates. Since the frequency is set to 10 s, the variation in the observed overhead is due to the number of changing paths. Note that the GS update only consists of changes occurred since the last update. The simulations show that the AeroRP overhead is much lower than AODV or DSDV since it does not transmit event-based updates.

Based on our examinations of the simulation trace data, the poor performance of AODV and DSDV is caused by the time scale on which they operate. In both cases they can take 30 s to 5 min to determine that a route has failed and reroute [90]. In an environment in which paths may only be stable for a few seconds, these protocols simply cannot keep up. While it is possible to minimize their route convergence time using modifications such as shorter update intervals and faster dead-link detection, this would inevitably lead to increased overhead.

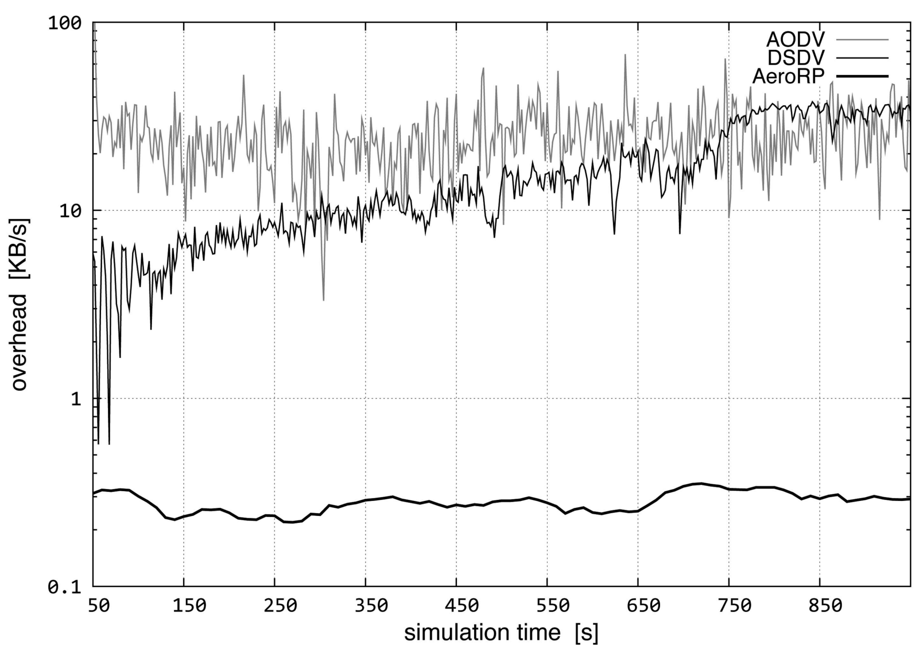


Fig. 19. Routing overhead for Mach 0.3 to 3.5.

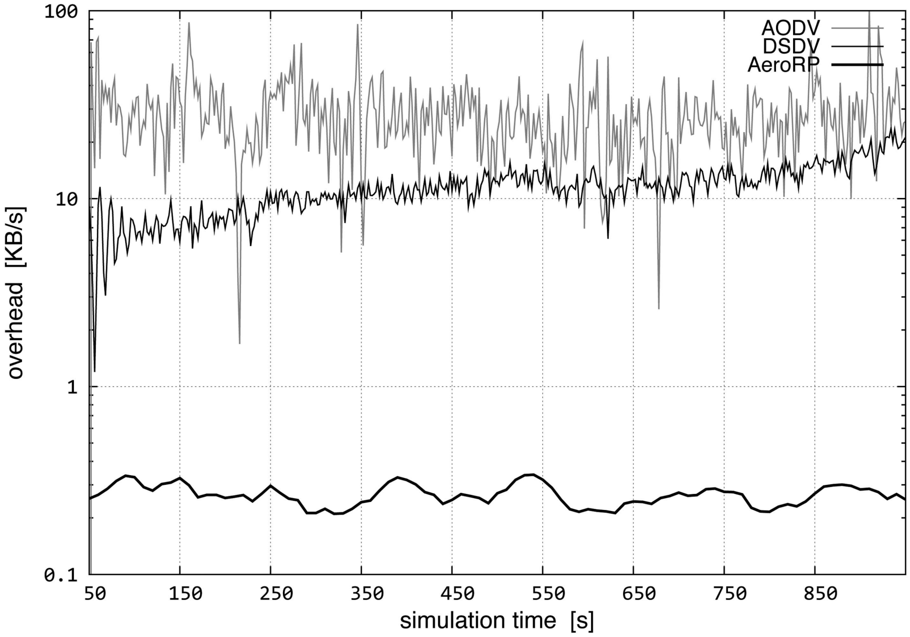


Fig. 20. Routing overhead for Mach 3.5.

VI. CONCLUSIONS AND FUTURE WORK

The existing Internet protocol architecture is not well suited for applications in highly-dynamic airborne tactical networks, which present unique challenges due to extreme mobility and limited bandwidth. Typical MANET routing protocols such as AODV, DSDV, DSR, and OLSR are not designed for topologies that are as dynamic as the ones found in aeronautical environments. In this paper, we describe a new protocol architecture that addresses these issues with domain-specific transport and network layers. It is observed that the exchange of information across layers provides significant benefit in the highly-mobile environment. We have developed domain-specific transport (AeroTP), network (AeroNP), and routing (AeroRP) protocols to leverage cross-layer information in optimizing end-to-end performance. By predicting when links will be available based on trajectory information, as well as actively listening for nearby nodes, AeroRP can send data opportunistically towards its destination and make much more efficient use of available network capacity. We performed simulations showing two orders of magnitude improvement in connection setup times when using AeroTP instead of TCP. We also performed simulations that show the new routing protocol performs significantly better with lower overhead than traditional proactive (DSDV) and reactive (AODV) MANET protocols in this environment. In the future we will perform more extensive AeroRP simulations with varying node densities, mobility models, etc. We will extend AeroTP with a full specification of the multipath erasure-coding mechanism. We are also working on additional simulations of AeroTP and will eventually be able to simulate the entire Aero protocol suite in ns-3 using a new 3D Gauss-Markov mobility model [91]. With the models refined iteratively through the simulation process, we will proceed to implementing prototypes of the entire Aero suite for field testing.

ACKNOWLEDGMENTS

We would like to acknowledge Kip Temple and the members of the iNET working group for discussions that led to this work.

REFERENCES

[1] Global information grid (GIG) overarching policy. Department of Defense Directive, Nov. 2003.

[2] Rohrer, J. P., Perrins, E., and Sterbenz, J. P. G. End-to-end disruption-tolerant transport protocol issues and design for airborne telemetry networks. In *Proceedings of the International Telemetering Conference*, San Diego, CA, Oct. 27–30, 2008.

[3] Jabbar, A., Perrins, E., and Sterbenz, J. P. G. A cross-layered protocol architecture for highly-dynamic multihop airborne telemetry networks. In *Proceedings of the International Telemetering Conference* (ITC), San Diego, CA, Oct. 27–30 2008.

[4] Rohrer, J. P., et al. Cross-layer architectural framework for highly-mobile multihop airborne telemetry networks. In *Proceedings of the IEEE Military Communications Conference* (MILCOM), San Diego, CA, Nov. 2008, 1–9.

[5] iNET System Architecture, version 2007.1 Central Test and Evaluation Investment Program (CTEIP), July 2007.

[6] iNET Needs Discernment Report, version 1.0 Central Test and Evaluation Investment Program (CTEIP), May 2004.

[7] iNET Technology Shortfalls Report, version 1.0 Central Test and Evaluation Investment Program (CTEIP), July 2004.

[8] Sterbenz, J. P. G., et al.
Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines.
Computer Networks: Special Issue on Resilient and Survivable Networks (COMNET), **54** (June 2010), 1245–1265.

[9] iNET Working Group
http://www.inetprogram.org.

[10] Abolhasan, M., Wysocki, T., and Dutkiewicz, E.
A review of routing protocols for mobile ad hoc networks.
Ad Hoc Networks, **2** (Jan. 2004), 1–22.

[11] Iannone, L., et al.
Cross-layer routing in wireless mesh networks.
In *Proceedings of the 1st International Wireless Communication Systems Symposium*, 2004, 319–323.

[12] Postel, J.
Transmission Control Protocol.
RFC Standard 793, Sept. 1981. Updated by RFCs 1122, 3168.

[13] Mathis, M.
TCP Selective Acknowledgment Options.
RFC 2018 (Proposed Standard), Oct. 1996.

[14] Allman, M., Paxson, V., and Stevens, W.
TCP Congestion Control.
RFC 2581 (Proposed Standard), Apr. 1999. Superseded by RFC 5681, updated by RFC 3390.

[15] Ramakrishnan, K., Floyd, S., and Black, D.
The Addition of Explicit Congestion Notification (ECN) to IP.
RFC 3168 (Proposed Standard), Sept. 2001.

[16] Braden, R.
T/TCP—TCP Extensions for Transactions Functional Specification
RFC 1644 (Experimental), July 1994.

[17] Jiong, L., Zhigang, C., and Junaid, K. M.
TP-satellite: A new transport protocol for satellite IP networks.
IEEE Transactions on Aerospace and Electronic Systems, **45** (Apr. 2009), 502–515.

[18] Allman, M., Glover, D., and Sanchez, L.
Enhancing TCP Over Satellite Channels using Standard Mechanisms.
RFC 2488 (Best Current Practice), Jan. 1999.

[19] Cola, T. D. and Marchese, M.
Performance analysis of data transfer protocols over space communications.
IEEE Transactions on Aerospace and Electronic Systems, **41** (Oct. 2005), 1200–1223.

[20] Elaasar, M. E., et al.
Satellite transport protocol handling bit corruption, handoff and limited connectivity.
IEEE Transactions on Aerospace and Electronic Systems, **41** (Apr. 2005), 489–502.

[21] Krishnan, R., et al.
Explicit transport error notification (ETEN) for error-prone wireless and satellite networks.
Computer Networks, **46**, 3 (Oct. 2004), 343–362.

[22] Balakrishnan, H., Katz, R. H., and Padmanbhan, V. N.
The effects of asymmetry on TCP performance.
Mobile Networks and Applications, **4**, 3 (Oct. 1999), 219–241.

[23] Allman, M., et al.
Ongoing TCP Research Related to Satellites.
RFC 2760 (Informational), Feb. 2000.

[24] Sterbenz, J. P. G.
Survivable mobile wireless networks: Issues, challenges, and research directions.
In *WiSE '02: Proceedings of the 3rd ACM Workshop on Wireless Security*, New York, 2002, 31–40.

[25] Postel, J.
User Datagram Protocol.
RFC Standard 768, Aug. 1980.

[26] Schulzrinne, H., et al.
RTP: A Transport Protocol for Real-Time Applications.
RFC Standard 3550, July 2003. Updated by RFC 5506.

[27] Durst, R. C., Miller, G. J., and Travis, E. J.
TCP extensions for space communications.
In *MobiCom '96: Proceedings of the 2nd Annual International Conference on Mobile Computing and Networking*, New York, Nov. 1996, 15–26.

[28] Wang, R. and Horan, S.
Protocol testing of SCPS-TP over NASA's ACTS asymmetric links.
IEEE Transactions on Aerospace and Electronic Systems, **45** (Apr. 2009), 790–798.

[29] Danzig, P., Liu, Z., and Yan, L.
An evaluation of TCP Vegas by live emulation, 1995.

[30] Hoe, J. C.
Improving the start-up behavior of a congestion control scheme for TCP.
In *SIGCOMM '96: Conference Proceedings on Applications, Technologies, Architectures, and Protocols for Computer Communications*, New York, 1996, 270–280.

[31] Floyd, S.
TCP and explicit congestion notification.
SIGCOMM Computer Communication Review, **24**, 5 (1994), 8–23.

[32] Partridge, C.
Gigabit Networking.
Reading, MA: Addison-Wesley Longman Publishing Co., 1994.

[33] Braden, R.
Requirements for Internet Hosts—Communication Layers.
RFC Standard 1122, Oct. 1989. Updated by RFCs 1349, 4379.

[34] Stevens, W.
TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms.
RFC 2001 (Proposed Standard), Jan. 1997. Superseded by RFC 2581.

[35] Jacobson, V. and Braden, R.
TCP Extensions for Long-Delay Paths.
RFC 1072, Oct. 1988. Superseded by RFCs 1323, 2018.

[36] Jacobson, V.
Compressing TCP/IP Headers for Low-Speed Serial Links.
RFC 1144 (Proposed Standard), Feb. 1990.

[37] Fox, R.
TCP Big Window and NAK Options.
RFC 1106, June 1989.

[38] Jacobson, V., Braden, R., and Borman, D.
TCP Extensions for High Performance.
RFC 1323 (Proposed Standard), May 1992.

[39] Moy, J.
OSPF Version 2.
RFC 2328 (Standard), Apr. 1998. Updated by RFC 5709.

[40] Malkin, G.
RIP Version 2.
RFC 2453 (Standard), Nov. 1998. Updated by RFC 4822.

[41] Rekhter, Y., Li, T., and Hares, S.
A Border Gateway Protocol 4 (BGP-4).
RFC 4271 (Draft Standard), Jan. 2006.

[42] Chiang, M., et al.
Layering as optimization decomposition: A mathematical theory of network architectures.
Proceedings of the IEEE, **95** (Jan. 2007), 255–312.

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

[44] Johnson, D., Hu, Y., and Maltz, D.
The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4.
RFC 4728 (Experimental), Feb. 2007.

[45] Perkins, C. E. and Bhagwat, P.
Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers.
In *Proceedings of the ACM SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications*, New York, 1994, 234–244.

[46] Clausen, T. and Jacquet, P.
Optimized Link State Routing Protocol (OLSR).
RFC 3626 (Experimental), Oct. 2003.

[47] Thedinger, T., Jabbar, A., and Sterbenz, J. P. G.
Store and haul with repeated controlled flooding.
Presented at the Second International IEEE Workshop on Mobile Computing and Networking Technologies (WMCNT), Moscow, Russia, Oct. 2010.

[48] Liao, W-H., Sheu, J-P., and and Tseng, Y-C.
GRID: A fully location-aware routing protocol for mobile ad hoc networks.
Telecommunication Systems, **18**, 1–3 (2001) 37–60.

[49] de la Fuente, M. and Ladiod, H.
A performance comparison of position-based routing approaches for mobile ad hoc networks.
In *Proceedings of the IEEE 66th Vehicular Technology Conference* (VTC2007-Fall), Baltimore, MD, Sept. 30–Oct. 3, 2007, 1–5.

[50] Galluccio, L.
A MAC/routing cross-layer approach to geographic forwarding in wireless sensor networks.
Ad Hoc Networks, **5**, 6, 2007, 872–884.

[51] Mauve, M., Widmer, A., and Hartenstein, H.
A survey on position-based routing in mobile ad hoc networks.
IEEE Network, **15**, 6 (2001), 30–39.

[52] Yuksel, M., Pradhan, R., and Kalyanaraman, S.
An implementation framework for trajectory-based routing in ad hoc networks.
Ad Hoc Networks, **4**, 1 (2006), 125–137.

[53] Iordanakis, M.
Ad-hoc routing protocol for aeronautical mobile ad-hoc networks.
In *Proceedings of the Fifth International Symposium on Communication Systems, Networks and Digital Signal Processing* (CSNDSP), 2006.

[54] Tchakountio, F. and Ramanathan, R.
Anticipatory routing for highly mobile endpoints.
In *Proceedings of the Sixth IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA), Washington, D.C., 2004, 94–101, 2004.

[55] Huang, H-Y., Khendry, A., and Robertazzi, T. G.
Layernet: A self-organizing protocol for small ad hoc networks.
IEEE Transactions on Aerospace and Electronic Systems, **38** (Apr. 2002), 378–387.

[56] Ahlswede, R., et al.
Network information flow.
IEEE Transactions on Information Theory, **46** (July 2000), 1204–1216.

[57] Sagduyu, Y. and Ephremides, A.
On joint MAC and network coding in wireless ad hoc networks.
IEEE Transactions on Information Theory, **53** (Oct. 2007), 3697–3713.

[58] Biswas, S. and Morris, R.
Opportunistic routing in multi-hop wireless networks.
SIGCOMM Computer Communication Review, **34**, 1 (2004), 69–74.

[59] Liu, H.
Opportunistic routing for wireless ad hoc and sensor networks: Present and future directions.
IEEE Communications Magazine, **47** (Dec. 2009), 103–109.

[60] Sanchez, J., Ruiz, P., and Marin-Perez, R.
Beacon-less geographic routing made practical: Challenges, design guidelines, and protocols.
IEEE Communications Magazine, **47** (Aug. 2009), 85–91.

[61] Blum, B., et al.
IGF: A state-free robust communication protocol for wireless sensor networks.
Department of Computer Science, University of Virginia, Charlottesville, Technical Report CS-2003-11, 2003.

[62] Sanchez, J., Marin-Perez, R., and Ruiz, P.
BOSS: Beacon-less on demand strategy for geographic routing in wireless sensor networks.
In *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems* (MASS), Oct. 2007, 1–10.

[63] Heissenbüttel, M., et al.
BLR: Beacon-less routing algorithm for mobile ad hoc networks.
Computer Communications, **27**, 11 (2004), 1076–1086.

[64] Handley, M., et al.
TCP Friendly Rate Control (TFRC): Protocol Specification.
RFC 3448 (Proposed Standard), Jan. 2003. Superseded by RFC 5348.

[65] Çetinkaya, E. K. and Sterbenz, J. P. G.
Aeronautical gateways: Supporting TCP/IP-based devices and applications over modern telemetry networks.
In *Proceedings of the International Telemetering Conference* (ITC), Las Vegas, NV, Oct. 2009.

[66] Border, J., et al.
Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations.
RFC 3135 (Informational), June 2001.

[67] Maltz, D. A. and Bhagwat, P.
TCP splice for application layer proxy performance.
Journal of High Speed Networks, **8**, 3 (1999), 225–240.

[68] Laprie, J-C.
Dependability: Basic concepts and terminology.
IFIP 10.4, Working Group on Dependable Computing and Fault Tolerance, draft, Aug. 1994.

[69] Mogul, J. and Deering, S.
Path MTU discovery.
RFC 1191 (Draft Standard), Nov. 1990.

[70] Sterbenz, J. P. and Parulkar, G. M.
Axon: Application-oriented lightweight transport protocol design.
In *Proceedings of the Tenth International Conference on Computer Communication* (ICCC), New Delhi, India, Nov. 1990, 379–387.

[71] Clark, D. D., Lambert, M. L., and Zhang, L.
NETBLT: A high throughput transport protocol.
SIGCOMM Computer Communication Review, **17** (Oct./Nov. 1987), 353–359.

[72] iNET TmNS Ground Segment Architecture, version 2007
Central Test and Evaluation Investment Program (CTEIP), July 2007.

[73] iNET TmNS Test Article Segment Architecture, version 2007
Central Test and Evaluation Investment Program (CTEIP), July 2007.

[74] Feldmeier, D.
An overview of the TP++ transport protocol project.
In A. N. Tantawy (Ed.), *High Performance Networks: Frontiers and Experience* (Kluwer International Series in Engineering and Computer Science), vol. 238, Boston, MA, Kluwer Academic, 1993, ch. 8.

[75] Burleigh, S., et al.
Delay-tolerant networking: An approach to interplanetary internet.
IEEE Communications Magazine, **41** (June 2003), 128–136.

[76] Scott, K. and Burleigh, S.
Bundle Protocol Specification.
RFC 5050 (Experimental), Nov. 2007.

[77] McAuley, A. J.
Reliable broadband communication using a burst erasure correcting code.
SIGCOMM Computer Communication Review, **20**, 4 (1990), 297–306.

[78] McQuillan, J., et al.
The new routing algorithm for the ARPANET.
IEEE Transactions on Communications, **28**, 5 (1980), 711–719.

[79] De Couto, D. S. J., et al.
Performance of multihop wireless networks: Shortest path is not enough.
In *Proceedings of the First Workshop on Hot Topics in Networks* (HotNets-I), Princeton, NJ, Oct. 2002.

[80] Heimlicher, S., et al.
On leveraging partial paths in partially-connected networks.
In *Proceeding of the 28th IEEE Conference on Computer Communications* (INFOCOM), Rio de Janeiro, Brazil, Apr. 2009.

[81] Ramakrishnan, K. K. and Jain, R.
A binary feedback scheme for congestion avoidance in computer networks.
ACM Transactions on Computer Systems, **8**, 2 (1990), 158–181.

[82] Mamun-Or-Rashid, M., et al.
Congestion avoidance and fair event detection in wireless sensor network.
IEICE Transactions on Communications, **E90-B**, 12 (2007), 3362–3372.

[83] Kelly, F., Maulloo, A., and Tan, D.
Rate control for communication networks: Shadow prices, proportional fairness and stability.
The Journal of the Operational Research Society, **49** (Mar. 1998), 237–252.

[84] Özveren, C., Simcoe, R., and Varghese, G.
Reliable and efficient hop-by-hop flow control.
In *SIGCOMM '94: Proceedings of the Conference on Communications Architectures, Protocols and Applications*, New York, 1994, 89–100.

[85] Wan, C., Eisenman, S., and Campbell, A.
CODA: congestion detection and avoidance in sensor networks.
In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, New York, 2003, 266–279.

[86] The ns-3 network simulator
http://www.nsnam.org, July 2009.

[87] The network simulator: ns-2
http://www.isi.edu/nsnam/ns/, Dec. 2007.

[88] Rohrer, J. P., et al.
ResiliNets models and tools for ns-3 network simulator.
http://wiki.ittc.ku.edu/resilinets/Models and Tools for ns-3 Network Simulator, Sept. 2010.

[89] Johnson, D. B. and Maltz, D. A.
Dynamic source routing in ad hoc wireless networks.
In T. Imielinski and H. F. Korth (Eds.), *Mobile Computing* (The Kluwer International Series in Engineering and Computer Science), vol. 353, Norwood, MA: Kluwer Academic Publishers, 1996, ch. 5, 153–181.

[90] Broch, J.
A performance comparison of multi-hop wireless ad hoc network routing protocols.
In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (MobiCom '98), Oct. 1998, 85–97.

[91] Broyles, D., Jabbar, A., and Sterbenz, J. P. G.
Design and analysis of a 3-D Gauss-Markov mobility model for highly-dynamic airborne networks.
In Proceedings of the International Telemetering Conference (ITC), San Diego, CA, Oct. 2010.

Justin P. Rohrer (S'03) received the B.S. degree in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2004. He is a Ph.D. candidate in the Department of Electrical Engineering and Computer Science at The University of Kansas, Lawrence.

From 1999 to 2004, he was with the Adirondack Are Network, Castleton, NY as a network engineer. He is currently a graduate research assistant at the KU Information & Telecommunication Technology Center (ITTC) and was an ITTC Graduate Fellow from 2004–2006. His research focus is on resilient and survivable transport protocols. His interests also include highly-dynamic mobile networks, simulating network disruptions, and developing the GpENI network testbed for the GENI program. Previous research has included weather disruption-tolerant mesh networks and free-space-optical metropolitan networks.

Mr. Rohrer received the best paper award at the International Telemetering Conference in 2008. He is a member of the IEEE Communications and Computer Societies, ACM SIGCOMM, Eta Kappa Nu, and is an officer of the Kansas City section of the IEEE Computer Society.

Abdul Jabbar (S'03) received his B.S. degree in electrical engineering from Osmania University, India, in 2001. He received his M.S. degree in 2004 and his Ph.D from the University of Kansas, Lawrence, in 2010, both in electrical engineering.

He currently works as an adjunct research associate at the KU Information & Telecommunication Technology Center. His interests include resilience and survivability, network algorithms, design and analysis of network architectures, topologies, and protocols, highly dynamic networks, wireless access, and future networks.

Dr. Jabbar is the recipient of the Moore award for best M.S. thesis and is a member of IEEE Communications Society, IEEE Computer Society, and the ACM Special Interest Group on Data Communications.

Egemen K. Çetinkaya (S'01) received the B.S. degree in electronics engineering from Uludag University, Bursa, Turkey, in 1999 and the M.S. degree in electrical engineering from the University of Missouri–Rolla in 2001. He is a Ph.D. student in the Department of Electrical Engineering and Computer Science at The University of Kansas, Lawrence.

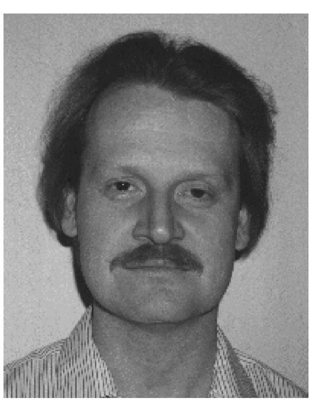
He held various positions at Sprint as a support system design engineer from 2001 until 2008. He is currently a graduate research assistant at the KU Information & Telecommunication Technology Center (ITTC). His research interests are in resilient networks.

Mr. Çetinkaya is a member of the IEEE Communications Society, ACM SIGCOMM, and Sigma Xi.

Erik Perrins (S'96—M'05—SM'06) received the B.S. (magna cum laude), M.S., and Ph.D. degrees from Brigham Young University, Provo, UT, in 1997, 1998, and 2005, respectively all in electrical engineering.

From 1998 to 2004, he was with Motorola, Inc., Schaumburg, IL, where he was engaged in research on land mobile radio products. Since August 2005, he has been with the Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, where he is currently an associate professor. Since 2004, he has also been an industry consultant on problems such as reduced-complexity receiver design and receiver synchronization. His current research interests include digital communication theory, advanced modulation techniques, synchronization, channel coding, and complexity reduction in receivers.

Dr. Perrins is an editor in the area of modulation theory for the *IEEE Transactions on Communications*. He is a member of the IEEE Communications Society and the IEEE Vehicular Technology Society.



James P. G. Sterbenz (S'87—M'91—SM'99) received a doctorate in computer science from Washington University in St. Louis, St. Louis, MO, in 1991, with undergraduate degrees in electrical engineering, computer science, and economics.

He is Associate Professor of Electrical Engineering & Computer Science and on staff at the Information & Telecommunication Technology Center at The University of Kansas, and is a Visiting Professor of Computing in InfoLab 21 at Lancaster University in the UK. He is director of the ResiliNets research group at KU, PI for the NSF-funded FIND Postmodern Internet Architecture project, lead PI for the GpENI (Great Plains Environment for Network Innovation) international GENI and FIRE testbed, co-I in the EU-funded FIRE ResumeNet project, and PI for the US DoD-funded highly-mobile airborne networking project. He has previously held senior staff and research management positions at BBN Technologies, GTE Laboratories, and IBM Research, where he has lead DARPA- and internally-funded research in mobile, wireless, active, and high-speed networks. His research interests include resilient, survivable, and disruption tolerant networking, future Internet architectures, active and programmable networks, and high-speed networking and systems.

Dr. Sterbenz has been program chair for IEEE GI, GBN, and HotI; IFIP IWSOS, PfHSN, and IWAN; and is on the editorial board of *IEEE Network*. He has been active in Science and Engineering Fair organisation and judging in Massachusetts and Kansas for middle and high-school students. He is principal author of the book *High-Speed Networking: A Systematic Approach to High-Bandwidth Low-Latency Communication.* He is a member of the IEEE, ACM, IET/IEE, and IEICE.