Network-Coded Connected Dominating Set Relaying for Airborne Tactical Networks

Leonid Veytser and Bow-Nan Cheng
MIT Lincoln Laboratory
244 Wood St. Lexington, MA 02420
{veytser, bcheng}@ll.mit.edu

ABSTRACT

In recent years, there's been a large push in the U.S. Department of Defense to provide greater bandwidth efficiency amidst congested spectrum. This is especially true in airborne tactical networks where transmit rates are limited and optimized for interference mitigation. Airborne tactical networks (ATNs) differ from many ground networks in that most of the traffic is broadcast and multicast. One method to more efficiently utilize the medium is to build a connected dominating set (CDS) backbone to reduce relay redundancy with overlapping nodes. While building a CDS has been shown to reduce retransmissions, thus saving bandwidth, backbone nodes can easily become congested. In this paper, we apply network coding to backbone CDS in airborne tactical networks to potentially mitigate the issues of congestion. Specifically, we implement network coded CDS (NCDS) in the Linux kernel and evaluate its performance compared to no coding under various relevant topologies. The results show that although NCDS can provide gains in tightly controlled topologies, the gains are severely limited in random and relevant ATN topologies.[1]

1. INTRODUCTION

In recent years, there's been a great push in the U.S. Department of Defense (DoD) to provide ubiquitous data connectivity out to the tactical edge. This net-centric vision brings with it increased bandwidth requirements amidst limited available spectrum and is especially challenging for airborne tactical networks where transmit rates are limited and optimized for interference mitigation [1]. Airborne tactical networks (ATNs) like Link 16 differ from many ground networks in that most of the traffic is broadcast and multicast in nature and intended for multiple receivers. It is therefore important to minimize the amount of redundant relaying while ensuring maximum coverage of transmissions.

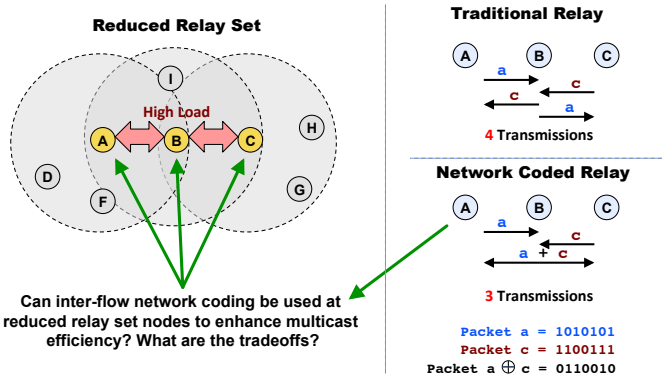

Figure 1: Applying inter-flow network coding to mitigate congestion on CDS backbones

One method to reduce bandwidth utilization and minimize wasted re-transmissions is to build a connected dominating set (CDS) backbone to relay multicast messages. CDS's are built by exchanging periodic hello messages with one or two hop information and selecting the relay nodes that cover the maximum amount of neighbors with one transmission. Although this method has been shown to be highly effective in reducing retransmissions, depending on how the CDS is formed, backbone nodes can exhibit high load and congestion as they act as relays for all nodes in the network. Figure 1 illustrates this problem. Nodes A, B, and C are selected as the CDS for the network. Instead of classical flood where all nodes relay traffic, as long as A, B, and C relay all broadcast/multicast messages, all other nodes in the network will receive a copy of the message. The result, however, is high load on nodes A, B, and C.

To reduce the load on the CDS backbone, we look to inter-flow network coding techniques. Figure 1 illustrates a basic example of inter-flow network coding using XOR to encode and decode packets. In traditional relay networks where the medium is shared, if node A has data to send to node C and node C has data to send to node A through a common relay node B, four transmissions are required to ensure delivery: Node A sends a to node B, node C sends c to node B, node B relays a to node C, and Node B relays c to node A. By applying inter-flow network coding at the relay node B, however, only three transmissions are required. In the case

[1]This work is sponsored by the Assistant Secretary of Defense - Research and Engineering (ASD-R&E) under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

AIRBORNE'14, August 11, 2014, Philadelphia, PA, USA.

ACM 978-1-4503-2985-9/14/08 ...$15.00.
http://dx.doi.org/10.1145/2636582.2636829.

of network coding, B knows that node A has packet a and node C has packet c and XORs a and c together to transmit. Because node A and C can both hear B's encoded message, and because they can simply XOR the coded message with the original message sent to recover the relayed message, only three transmissions are required. By applying network coding to CDS backbone nodes, significant load reduction can potentially be realized.

In this paper, we present a network coded connected dominating set (NCDS) approach to reduce broadcast/multicast network load on highly congested CDS backbones. We evaluate the approach on several relevant airborne tactical network topologies and draw insights on feasibility and gains of the approach. Key contributions of the paper include:

- Design and implementation of a Linux kernel inter-flow NCDS module
- Performance evaluation in several random and relevant airborne tactical network (ATN) topologies
- Implications of using NCDS in ATNs

The rest of the paper is organized as follows: Section 2 surveys inter-flow network coding techniques for broadcast wireless transmissions describing basic functionality, background and algorithms. Next, Section 3 overviews the NCDS approach including design considerations and the Linux kernel implementation of NCDS. Section 4 provides a basic evaluation of the implementation under several random as well as ATN relevant topologies while Section 5 discusses findings and relevant lessons learned. Finally, Section 6 concludes the paper.

2. NCDS RELATED WORK

Although network coding has been researched and applied to wired networks to facilitate efficient multicast dissemination as early as 2000 [2], it was first applied practically to wireless networks in COPE [3]. In COPE, the broadcast nature of 802.11 networks was leveraged to XOR together several *unicast* flows from various sources and sinks in a wireless mesh network to reduce bandwidth. The authors showed that by opportunistically listening to traffic being sent over the air, understanding neighborhood information, and intelligently mixing packets from different unicast flows, COPE's throughput gains varied from 1.33 to 1.6 for various fixed topologies. Although the concept presented in COPE is interesting, due to the dominant nature of multicast traffic on airborne tactical networks, this approach cannot be directly leveraged.

In [4], Li Li et. al. extend the concept of COPE to wireless broadcast transmissions, extending the XOR coding concept to a Reed-Solomon approach. They show that applying a simple XOR-based coding scheme algorithm to broadcast traffic can solicit a 45% gain compared to non-coding approaches. With the Reed-Solomon-based approach, the gains increase to 61%.

Wang et. al [5] proposed calculating the maximum independent set (MIS) and applying network coding to only those nodes in the relay set. Their work showed that NCDS provides up to 161% gains compared to *classical flooding* and 37% gains compared to CDS-based broadcasting without network coding. Although the results are promising, the work focused on tight coupling between CDS algorithm and network coding strategy and measured energy reduction in simulation. Our work examines packet delivery and network load on live networks, decoupling CDS selection algorithms from coding scheme. We leverage widely available routing protocols like OSPF-MDR and NHDP, which perform CDS calculation to select CDS and perform coding in a greedy manner.

In [6], Veytser et. al. presented a Linux kernel implementation of broadcast inter-flow network coding. The evaluation focused on simple fixed topologies and significant gains were shown to be achieved. In the work, nodes were manually assigned to code and relay. We extend this work both in implementation (dynamically electing CDS and coding over CDS only and not all nodes) as well as evaluation (examining random and relevant ATN topologies). Additionally, we introduce network coded connected dominating sets (NCDS) to mitigate issues with CDS congestion.

3. NCDS OVERVIEW

There are two major components in the design of network-coded connected dominating sets: the CDS selection algorithm and the network coding technique. Our design decouples the two components so that multiple CDS selection algorithms can be tested against multiple network coding techniques. In this section, we overview the CDS selection technique and the network coding algorithm as well as the kernel implementation of NCDS.

The CDS selection algorithm operates in the control plane, actively determining which nodes are elected as a relay. In many cases, CDS algorithms require the exchange of 2-hop neighbor information which is often a byproduct of the routing protocol discovery. Protocols like Neighborhood Discovery Protocol (NHDP) [7] actively include 1-hop neighbors in their *hello* messages, enabling all neighbors to acquire 2-hop topology. Once 2-hop neighborhood information is discovered, CDS selection algorithms can be applied to self-elect and determine if the current node is a relay. If the node is a relay, a flag is set to enable relaying and if not, the flag is unset to disable relaying.

In our tests, we used the essential connected dominating set (E-CDS) algorithm described in [8, 9] to elect the CDS nodes. E-CDS relays are "self-elected" using a Router Identifier (Router ID) and an optional nodal metric (Router Priority). Each node must have a consistent view of their Router IDs and priorities. The E-CDS self-election process follows two simple steps: If the node has a higher Router ID and/or Router Priority than all of its symmetric neighbors, then it elects itself to act as a forwarded for all received multicast packets. Otherwise, if there does not exist a path from the neighbor with the largest Router ID or Router Priority to any other neighbor, via neighbors with larger values of Router IDs or Router Priorities, then it elects itself as a relay node. The E-CDS algorithm is used in OSPF-MDR [8] to build a CDS backbone to relay link state advertisements and has been shown to scale fairly well.

The network coding technique used is similar to [6] in that relay nodes search through the queue to determine coding opportunities and uses XOR functions to encode packets together greedily. To determine which packets to encode, it is important to know which neighbors can successfully decode packets and maximize the opportunities to decode. Two methods to determine whether a neighbor has packets that can be used to decode a message are 1) if a neighbor is the

previous hop of the received packet, and 2) if a neighbor is also a neighbor of another neighbor. In a broadcast medium, neighbors of neighbors should have a high probability of receiving all packets sent by the neighbor. In our work, we assume that all neighbors of our neighbors have successfully received packets we have received and the NCDS module maintains a list of all 2-hop neighbors as determined by the routing protocol.

Packet encoding follows a greedy algorithm to find coding opportunities by searching among the packets enqueued in the local FIFO queue. For each sequential packet found in the queue, a check is performed to see whether all of the sender's neighbors can decode the resulting packet if it is combined with an already existing set of encoded packets. To ensure that all the neighbors can decode a newly encoded packet, the reception tables in the packet database are used to verify that all of the sender's neighbors have already received at least *n - 1* of these packets with high probability. The reception probabilities can be calculated from link quality information passed down from higher layer protocols like NHDP, OSPF, etc. If this decoding rule is not satisfied, then some neighbors will be unable to decode the packet.

If a packet is deemed safe to encode with the existing set of encoded packets, it is removed from the queue. The packet is then handed to the encoding procedure. To encode the packet with one or more other packets, the encoding procedure XORs them together to create the payload of the encoded packet. If the packets are of different length, the smaller packets are padded with zeros. The network coding header is modified to reflect the addition of another native packet to the encoded payload. After all the queues have been traversed, the resulting encoded packet is sent out. If no coding opportunities are found the native packet is transmitted without delay.

When an encoded packet is received by a node, the decoder performs a lookup in the packet database for all the packet ID's listed in the network coding header. If it can find *n - 1* packets (where *n* is the number of packets encoded) the packet can be successfully decoded. To decode the packet, the decoder XORs the payload of the encoded packet with the *n - 1* native packets. It then removes the network coding header from the packet and modifies its EtherType to reflect the protocol contained in the decoded packet (e.g. IPv4 or IPv6). If necessary, the decoder removes the zero padding and adjusts the length of the packet. It then stores the resulting decoded packet in the packet database to help with future decodings and passes it up the networking stack.

3.1 NCDS Kernel Implementation

In this section, we overview the key components of the NCDS kernel implementation. The NCDS implementation builds off the broadcast network coding kernel implementation [6], leveraging CDS calculations to make forwarding decisions. The Linux kernel NCDS implementation operates as a virtual device attached to a physical interface. Figure 2 illustrates the major functions and a sample packet flow through the system. As shown in Figure 2, the module enslaves a physical device and exposes a virtual network device that routing tables and processes can use to send and receive data (ncX). The control interface allows user-space programs to setup and configure the virtual network coded devices as well as modify various NCDS module state. Processes like OLSR [10], OSPF [8], and NHDP [7] are expected

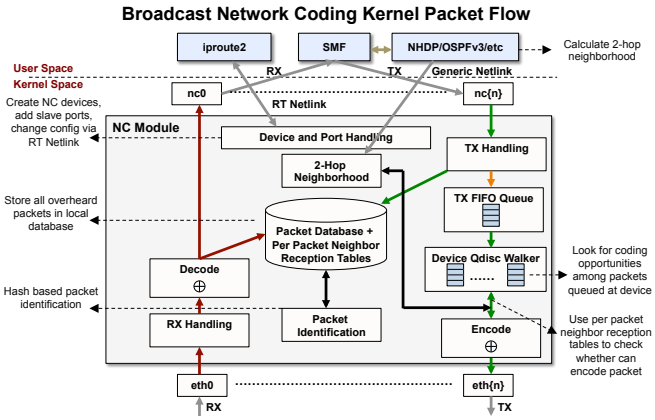

Figure 2: Packet Flow Diagram for NCDS Module

to dynamically inject 2-hop neighbor and link quality information into the NCDS module so the module knows which 2-hop neighbors are available and can potentially overhear transmitted packets.

When a packet is received from the physical interface (eth0) and delivered to the NCDS module, the packet is first decoded by XOR-ing it with packets that have already been overheard in the packet database. The resultant packet is hashed and stored in the database to potentially be used in another decoding. After decoding the packet and storing a copy for future use, the packet is delivered up the stack.

Outgoing packets received from higher layers intended for transmission are first placed in a local FIFO queue and a copy placed in the packet database. The FIFO queue is then serviced by a worker thread. For each packet leaving the FIFO queue, the module's FIFO queue and the device's queues are traversed to check whether the packet being sent out can be encoded with one or more other packets that are waiting in the queues. If candidate packets for encoding are found, they are removed from their places in the queues and handed to the encoding module. After the packets are encoded by XOR-ing them together, the resulting packet is sent to the slave physical device to be transmitted.

The NCDS module handles the receiving of packets that are received on its slave device and sending of the packets that are given to it to send by the Linux kernel networking stack. It does not perform any packet forwarding. An external process such as Simplified Multicast Forwarding (SMF) [9] can be used to forward multicast and broadcast packets over the ncX interfaces.

To facilitate network coding only at CDS nodes the module relies on the forwarding mechanism (SMF) to determine whether the node should transmit or not. SMF relies on a routing protocol like OLSR, NHDP, or OSPF-MDR to send hello messages and calculate CDS and flags (through shared memory) whether a particular node should relay or not. When a multicast or broadcast packet arrives at the SMF process to relay, SMF will read the shared memory and determine whether to relay or not based on OLSR, NHDP, or OSPF-MDR's CDS algorithm. If it is a relay, it will send the packet down the NC device which will apply network coding on the outgoing packet. If the node is not elected as a relay (not in CDS), the packet will not be sent. In this way, the implementation separates control (CDS al-

gorithm decision) from data flow and allows multiple CDS algorithms and NCDS coding mechanisms to be employed independently. More details of the implementation can be found in [6]

4. PERFORMANCE EVALUATION

In this section, we provide a performance evaluation of the NCDS implementation under various random and ATN-relevant topologies [11]. We compare classical flooding (CF), classical flooding with network coding applied on all nodes (NCF), CDS reduced flooding based on NHDP's implementation of essential connected dominating set algorithm (E-CDS), and the network-coded CDS algorithm (NCDS). Specifically, we examine aggregate received data rate with various number of traffic flows under 10, 20, 30 node random topologies and a 9 node airborne tactical topology. Aggregate received data rate includes all data intended for the node as well as relay data. For example, if a node receives 1 Mbps and is the intended recipient, but also receives 1 Mbps to relay, the aggregate received data rate will be 2 Mbps. It does not include duplicate packets received as well data sourced from the node. All nodes are running SMF for the forwarding engine with hash-based duplicate packet detection (DPD) enabled.

To evaluate the NCDS implementation, Common Open Research Emulator (CORE) [12] was used. CORE emulates each node using Linux containers with each node having its own separate network namespace. The topologies are generated using CORE's basic range model with 275 meter transmission range, 0 ms jitter, 20 ms delay, and 0% error. The topologies are static and the links in the topologies are stable. Each node's interface is configured with the token bucket (TBF) queueing discipline at 1 Mbps with buffer and limit set to 1600 bytes and 15,000 bytes respectively, effectively making that data rate the transmit capacity of each node. All the flows are generated using MGEN with packet sizes of 1300 bytes at 1 Mbps with a periodic interval. Flows are chosen with random multicast source and all destinations and last for the duration of the experiment. Since topology change due to mobility is fairly rare in airborne tactical networks [1], we simplify the tests by keeping all nodes fixed and examining the effects of NCDS on these static topologies. Future work includes adding mobility and aircraft body blockage. In the following subsections, we present results of tests with random and ATN-relevant topologies. All tests were averaged over 6 runs.

4.1 Random Topology Tests

Our initial tests focused on evaluating NCDS under various random topologies and traffic loads. Random topologies were generated in a 1500 meter x 1125 meter rectangle. Node transmission radius was limited to 275 meters and full connectivity checked prior to running the tests. In the tests, we varied the number of random source 1 Mbps multicast flows from 2 to the number of nodes in the network N and measured the aggregate data rate received by all nodes.

Figure 3 shows the aggregate data rate received as a function of the number of number of senders. As expected, pure CF delivered the lowest amount of traffic as the numbers of senders increased due to network congestion. With fewer senders (less than 15), NCF performed well (19-44% more data delivered than CDS) because many nodes can aid in coding the data. As the number of flows increase, however,

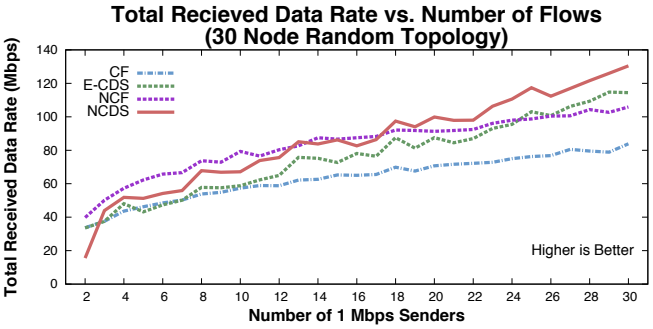

Figure 3: Total data rate received vs. number of senders

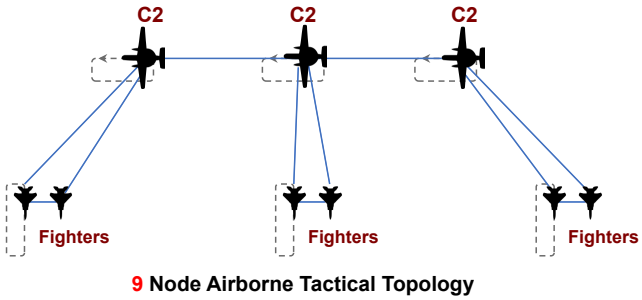

Figure 4: 9 node airborne tactical network topology

both E-CDS and NCDS begin to deliver more data than the other approaches (NCDS 11-23% more than NCF) . The reason for this is that with E-CDS, fewer nodes are relaying, causing much fewer transmissions overall on the network and less congestion. The gains of NCDS over E-CDS are fairly consistent across the range of senders (10-19% better) as NCDS builds off of E-CDS's savings. Results for 10 and 20 randomly placed node topologies shows a similar effect.

4.2 Airborne Tactical Network Topology Tests

In this section, we evaluate NCDS over a representative airborne tactical network topology. Figure 4 illustrates the topology. There are 3 command and control (C2) aircraft and 3 groups of 2-ship fighters. The C2 aircraft act as the relay and backbone for the fighters and all nodes want to obtain all the multicast traffic. As with the random topology tests, we increase the number of random source 1 Mbps flows from 2 to 9 and measure the total received data rate.

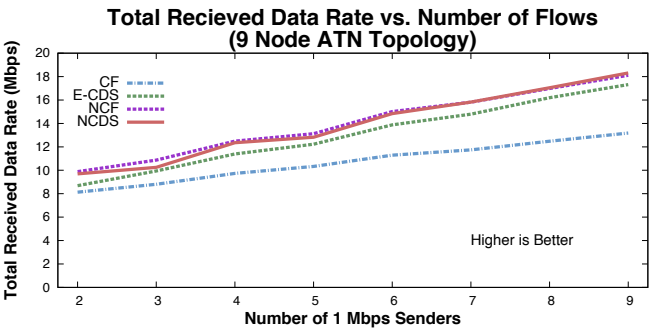

Figure 5: Total rate received vs. number of senders

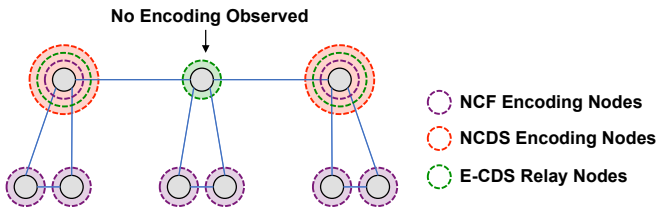

Figure 6: Breakdown of encoders and relayers in 9 node ATN topology

Figure 5 shows the total received data rate as compared to the number of senders in the ATN topology. CF, as expected, delivers the least amount of data because it overruns the network with data. Oddly, in the ATN topology, E-CDS performers worse than both NCF and NCDS (5-14% lower data rate delivered). In fact, NCF seems to perform as good or better than NCDS in almost all cases. To understand what's going on in this test case, it is important to understand which nodes are acting as relays under E-CDS and which nodes are encoding packets under NCF and NCDS. Figure 6 highlights these cases. As can be seen, with NCF, almost all nodes act as encoders and can combine packets as needed, resulting in significant bandwidth savings. With NCDS, however, only two nodes encode packets, resulting in minimal gain compared to E-CDS.

One interesting observation is that the center node (shown in Figure 6) performs no encoding. The greedy network coding algorithm employed attempts to ensure that all neighbors can decode packets successfully. Because there are 3 disjoint sets of neighbors, the center node cannot assume any neighbor has overheard the other neighbor's transmissions. As a result, it sends only unencoded packets. Additional techniques can potentially be explored such as creating two encoded packets instead of one to satisfy the disjoint neighbor set [13]. These concepts are not explored in this paper.

5. DISCUSSION

Throughout the performance testing, there were several lessons learned and interesting observations. First, because network coding takes a packet off the egress queue and codes it with an outgoing packet, encoding will never take place until queues start becoming backlogged. For light loads, the gains of intra-flow network coding are not seen. Second, in general, N disjoint neighbor sets require $N-1$ coded packets to ensure all neighbors can decode the message. In the case of the airborne tactical scenario, the center node never encoded any packets because it had 3 disjoint neighbor sets and could only send 1 coded packet at a time. Because it could not ensure all neighbors could decode the packet, it only send unencoded packets, leading to lack of gains.

Another observation is that under some topologies, running either E-CDS or NCF was sufficient to provide almost all the gains. The addition of network coding on CDS relay nodes only provided marginal benefit. In fact, the more minimal the CDS election algorithm, the less effective NCDS becomes. CDS selection algorithms attempt to minimize the overlapping transmissions. In contrast, broadcast network coding requires overlap to ensure neighbors have overhead packets to be able to successfully decode. The opposite goals of CDS and network coding can lead to little to no gains under some topologies and with some CDS algorithms.

6. CONCLUSION AND FUTURE WORK

In recent years, there has been a large push in the DoD and in particular, airborne tactical networks, to maximize bandwidth efficiency and end-to-end reliability. Traffic on airborne tactical networks are predominantly multicast. By leveraging technologies like connected dominating set (CDS) theory, reductions in relay traffic can be achieved. One issue with using CDS selection is that CDS backbone nodes suffer from high load. In this paper, we examine leveraging network coding to reduce load from broadcast/multicast traffic flow on CDS backbones. We introduce and implement a network-coded connected dominating set (NCDS) kernel implementation and quantify its performance on several random and ATN-relevant topologies.

From the results, it can be seen that applying network coding to multicast flooding can provide significant gains to data delivery. However, the gains by applying network coding to CDS nodes in airborne topologies only give minor gains over pure CDS and little to no gains over NCF. The reason is that ATN topologies are oriented in a way that coding opportunities are not available on CDS backbones. Additionally, because CDS theory attempts to minimize node overlap, and network coding thrives on node overlap, the two techniques are diametrically opposed. Future work includes evaluating larger ATN topologies and introducing link loss and mobility.

7. REFERENCES

[1] B.-N. Cheng et al., "Design Considerations for Next-Generation Airborne Tactical Networks," *IEEE Communications Magazine*, May 2014.

[2] R. Ahlswede et al., "Network information flow," in *IEEE Transactions on Information Theory*, 2000.

[3] S. Katti et al., "XORs in The Air: Practical Wireless Network Coding," in *ACM SIGCOMM*, 2005.

[4] L. Li et al., "Network coding-based broadcast in mobile ad hoc networks," in *IEEE INFOCOM*, 2007.

[5] S. Wang et al., "Energy Efficient Broadcasting Using Network Coding Aware Protocol in Wireless Ad hoc Network," in *IEEE ICC*, 2011.

[6] L. Veytser et al., "A Linux Kernel Implementation of Broadcast Interflow Network Coding," in *IEEE MILCOM*, 2013.

[7] T. Clausen et al., "Mobile ad hoc network (manet) neighborhood discovery protocol (nhdp)," IETF, RFC 6130, 2011.

[8] R. Ogier et al., "Mobile Ad Hoc Network (MANET) Extension of OSPF Using Connected Dominating Set (CDS) Flooding," IETF RFC 5614, 2009.

[9] J. Macker *et al.*, "Simplified multicast forwarding," Internet Engineering Task Force, RFC 6621, 2012.

[10] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," IETF RFC 3626, 2007.

[11] B.-N. Cheng and S. Moore, "An Evaluation of MANET Routing Protocols on Airborne Tactical Networks," in *IEEE Military Communications Conference, MILCOM 2012*, October 2012.

[12] J. Ahrenholz, "Comparison of CORE Network Emulation Platforms," in *IEEE MILCOM*, 2010.

[13] N. Jones et al., "Optimal Routing and Scheduling for a Simple Network Coding Scheme," in *IEEE INFOCOM*, 2012.