Large Scale Over-the-Air Testing of Group Centric Networking

Logan Mercer, Greg Kuperman, Andrew Hunter, Brian Proulx
MIT Lincoln Laboratory
Lexington, MA, USA 02420
{logan.mercer, gkuperman, andrew.hunter, brian.proulx}@ll.mit.edu

Abstract—**This paper presents experimental verification of the performance of Group Centric Networking (GCN), a networking protocol developed for robust and scalable communications in lossy networks where users are localized to geographic areas, such as military tactical networks. Initial simulations in NS3 showed that GCN offers high delivery with low network overhead in the presence of high packet loss and high mobility. We extend the investigation to verify GCN's performance in actual over-the-air experimentation.**

In the experiments, we deployed GCN on a 90-node Android phone test bed distributed across an office building, allowing us to evaluate its performance over-the-air on real-world hardware. GCN's performance is compared against multiple popular wireless routing protocols, which we also run over-the-air. These tests yield two notable results: (1) the seemingly benign environment of an office is in fact quite lossy, with high packet error rates between users that are geographically close to one another, and (2) that GCN does indeed offer high delivery with low network overhead, which is in contrast to traditional wireless routing schemes that offer either high delivery or low overhead, or sometimes neither.

I. Introduction

With the desire to connect every soldier, vehicle, and sensor together via a battlefield intranet come numerous networking challenges. Commercial networks enjoy fixed infrastructure with stable high-capacity links, as well as low mobility of users. These conditions enable a variety of network traffic patterns with little penalty for inefficient network use. Military tactical networks, however, operate under very different conditions: pre-existing infrastructure does not exist in the battlefield, connections are often low-rate and lossy, and soldiers and vehicles move rapidly in an ever-changing hostile landscape. Due to these fundamental challenges, robustly connecting users together in military tactical networks is still an open problem.

A similar problem is developing in the commercial world. The Internet of Things (IoT) [1] is the idea that every device, from the refrigerator to medical devices to automobiles, will connect to one another. In a given location (house, factory, etc.), we expect 10s to 100s of devices to communicate with one another. Under this model, the amount of wireless traffic will skyrocket, and a single WiFi access point may not be able to process and coordinate all of the incoming data. Furthermore, much of this data is potentially destined for other devices operating in the same area, making the access point an unneeded bottle neck; those users can more efficiently communicate directly between themselves. Current wireless ad-hoc routing schemes unfortunately do not scale well and have poor performance overall [2]. Because of this, there has been a large push to develop new schemes that can meet the challenges of the emerging IoT paradigm [3].

This work is sponsored by the Defense Advanced Research Program Agency via Air Force contract #FA8721-05-C-0002. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Approved for public release; distribution unlimited.

Tactical networks and the Internet of Things highlight the new challenges that need to be addressed by future protocols; in particular, handling the amount of traffic, the number of devices, and the ad-hoc nature of the network. Group Centric Networking (GCN) is a proposed networking protocol that addresses challenges specific to military networks and IoT [4]. GCN operates by efficiently connecting users that share common interests within a certain geographic area. GCN moves from the address-centric model of networking, where devices send messages to explicit addresses in a client-server model, to a group-centric model where users that share a common set of interests communicate in a collaborative nature. This exploits the nature of the underlying network. For instance, in military networks, most communications are held within a local area and tend to exhibit collaborative traffic patterns (e.g., situational awareness, sensor fusion) as opposed to the client-server nature of more traditional networks [5, 6]. GCN offers the ability to scale to a large number of nodes because it efficiently connects interested users without flooding unneeded control information. Consider in the Internet of Things where an array of sensors are trying to communicate to some centralized controller. GCN allows these users to efficiently find one another and then to robustly exchange data.

When examining the success of a routing protocol, one must examine both the protocol's success at delivering packets and how many resources the protocols uses in the attempt. In highly dynamic networks, many wireless routing protocols send high amounts of control information while achieving only low delivery rates. Conversely, flooding protocols send little to no control information and can achieve high packet delivery, but at the expense of too much message redundancy that overloads the network. GCN aims to provide high delivery with low network load in lossy environments by balancing redundancy and control and taking advantage of the wireless medium.

Prior to this article, GCN had only been evaluated in simulation. Here we explore the real-world performance of GCN on actual hardware, and compare its performance against several other ad-hoc routing protocols. We conduct our tests over-the-air on a 90-node Android phone test bed in an office environment. We find that GCN outperforms wireless unicast routing protocols by wide margins with respect to delivery ratio, and offers delivery rates comparable to flooding protocols while using as little as a third of the overall network resources. In addition, GCN also achieves higher delivery rates than flooding protocols in certain scenarios due to the flooding approach completely overwhelming network capacity. We also find that an office environment is in fact quite lossy, with users in close distance to one another experiencing high packet loss.

The rest of the paper is organized as follows. In Section II, GCN is introduced and a brief summary of routing protocols is offered. The test bed used for the experiments is described in Section III. A discussion of link quality is discussed in Section IV, and test results are presented in Section V.

II. Review of GCN and Other Wireless Routing Protocols

In this section, we first provide a brief survey of the various routing schemes that we compare GCN against. We then present a brief overview of GCN.

A. Wireless Routing Protocols

In order to test the effectiveness of GCN, we compare against three other wireless routing protocols.

1) Optimized Link State Routing (OLSR): OLSR is a proactive routing protocol where routes are continuously maintained between all users [7]. In OLSR, every few seconds each node sends a neighbor list to its neighbors and a global topology view to the entire network. OLSR employs multi-point relays (MPRs) to minimize the number of nodes retransmitting messages. With a local and global view of the network, a node can determine shortest paths between itself and all other users.

2) BABEL: Babel is a loop-avoiding distance-vector routing protocol based on Destination-Sequenced Distance Vector Routing (DSDV) [8]. It is a proactive routing protocol that uses the distributed Bellman-Ford algorithm to guarantee that no routing cycle is established between nodes. In the Babel protocol, nodes maintain several data structures to calculate paths. In particular, nodes keep a list of their neighbors and make explicit requests for routing information.

3) SMF: Simple Multicast Forwarding (SMF) is a basic flooding protocol [9]. In SMF, a message is transmitted with some time-to-live (TTL)[1], and the message is rebroadcast by neighboring nodes. Each rebroadcast decrements the TTL until the TTL reaches zero. There is no control messaging in SMF – if a device hears a valid message with a nonzero TTL it will rebroadcast that message. The protocol uses duplicate detection to try to limit the number of packets transmitted, which works by checking received packets for the ID of the original sender as well as a packet sequence number. If a device has seen a message from that source with that sequence number before then it must be a duplicate and is thrown out. SMF seeks to achieve a high delivery rate at the cost of enormous data overhead.

[1]Time-to-live is a method to prevent data from indefinitely circulating a network. It is implemented as a counter attached to a packet that is decremented every time a message is sent. TTL is also known as a hop limit.

B. Group Centric Networking

Group Centric Networking is a networking protocol designed to support groups of devices in a local region [4]. It attempts to use the wireless medium to broadcast minimal control information, which allows it to scale to very large numbers of nodes. Both unicast and multicast messaging are supported. GCN is organized around the concept of a *group*, which is a collection of devices in some geographic area that are interested in each others traffic. Groups can be formed on arbitrary subsets of nodes, and one node may be in many groups. To efficiently and robustly disseminate data, three major mechanisms are employed: Group Discovery, Tunable Resiliency, and Targeted Flooding. Group Discovery connects interested users together, and Tunable Resiliency and Targeted Flooding allow for robust traffic flow between these users. A brief description of each of the major mechanisms are presented; further details and analysis can be found in [4].

1) Group Discovery: The goal of the group discovery algorithm is to find group nodes without globally flooding control messages. To facilitate this, GCN uses discovery regeneration, an adaptation of the familiar Time-to-Live (TTL) approach. A node begins group discovery by sending out a message with a low TTL. Every time a group node hears a group discovery message it refreshes the TTL, effectively *regenerating* the discovery message, while non-group nodes only decrement the TTL. Thus the discovery message will only be heard in local regions near other group nodes.

Data relays are elected by acknowledgement messages (ACK) sent in response to discovery messages. All nodes in between group nodes that receive an ACK are elected as data relays. Duplicate detection is used to ensure discovery messages are broadcast only once. Once a node is elected as a relay it acts as a relay for the entire group, not just the nodes it originally acknowledged. Nodes do not need to store any information about neighbors or to whom they sent ACKs – relays act as relays for the entire group, not only for a particular node. At the end of the group discovery algorithm each group node has a path to any other group node, thus enabling a one-to-many (multicast) traffic pattern.

2) Tunable Resiliency: Tunable resiliency is a mechanism to activate additional relays during the group discovery process. Group discovery will create a minimal spanning tree between group nodes, but a single failure will potentially disconnect the network. To increase resiliency, group discovery is extended to have devices probabilistically self-select as data relays.

When a node enters the network it enters with a preset resiliency parameter R, with group discovery messages containing this parameter. This parameter represents the number of neighbors this node desires to have as relays. After group discovery, one node is selected as an *obligate* relay. An additional $R-1$ nodes need to be activated as relays. If a node has N neighbors, it sets the probability that a neighbor self-selects to become a relay to $\frac{R-1}{N-1}$. Each neighbor opts into becoming a relay with that probability, so a node has on average R relays in its neighborhood.

GCN's Tunable Resiliency allows the network to increase redundancy by allowing nodes to self-select as redundant relays and thus perform well in lossy networks in an efficient manner.

3) Targeted Flooding: We have so far discussed Group Centric Networking's one-to-many communications, but GCN also includes an effective way to send data one-to-one (unicast) via a "targeted flooding" mechanism. Targeted flooding uses distance information gathered from overheard packets to create a distributed gradient field towards each of the group members. Then, when a user wants to send data to another it simply forwards the packet in the direction of the distance gradient. GCN does not record any information about particular links or neighbors, it simply maintains this distance gradient.

To increase the resiliency of Targeted Flooding, GCN can opt to send duplicate packets towards less optimal gradient routes. With no resiliency GCN only sends a single packet along the most likely distance gradient, but with higher resiliency other packets will be sent to try to reach the destination. To create a many-to-one traffic pattern (such as a set of sensors), each of the many nodes can establish a one-to-one connection with a sink node.

C. Recap of Simulation Results for GCN

As mentioned previously, extensive simulation results were performed for GCN in [4]. In particular, GCN was implemented in Network Simulator 3 (NS3), with comparisons to SMF and Ad-Hoc On-Demand Distance Vector (AODV) routing [10] (which is another commonly used wireless routing protocol). Networks of 50 and 100 users were compared, and performance was examined under realistic packet error rate curves.

In those simulation results, GCN significantly outperformed AODV with respect to delivery and outperformed SMF with respect to efficiency. In particular, in high loss environments, GCN delivered 97% of traffic while AODV delivered between 6% and 12%. Since SMF is a flooding protocol, it was able to deliver close to 100% of packets in this scenario; however, GCN transmitted an order of magnitude fewer packets over-the-air to achieve a similar delivery rate.

III. Test Setup

In order to compare GCN against the other routing protocols in real networks on real devices, we compiled and tested OLSR, Babel, GCN, and SMF on a network of 90 Android phones in an office environment. The test setup is described in this section.

A. Android Phones

The Android phones are Samsung Galaxy S4 running Cyanogenmod 10.2. For the network, we use 802.11ac WiFi running in the 5GHz band and using a transmit power of 1 mW. All transmissions occur via 802.11ac broadcast mode, which has no acknowledgement packets and runs at a fixed rate of 6 Mbps [11]. We purposely choose to transmit via broadcast mode in order to avoid any 802.11-specific MAC layer features that may not exist in other networks, such as MAC layer retransmits and rate adaptation. Furthermore, by limiting the MAC to be a simple broadcast, we are better able to isolate and evaluate the network layer protocols.

In order to generate traffic for OLSR and Babel, the devices run Multi-Generator (MGEN) [12]. GCN is built with its own traffic generation for testing purposes.

B. Topology

We tested the routing protocols over several topologies within an office building with varied layouts and densities to understand their performance. The office floor plan is shown in Figure 1.

Phones are distributed with approximately 1 to 3 phones per room, resulting in a large network diameter. In this topology there can be many paths from one node to another. In order to vary network density, we test this topology with 30, 60, and 90 nodes spread over the same area. This topology is meant to be stressing and is realistic in that it is very similar to the actual distribution of smartphones found in those offices during normal working hours.

C. Traffic Patterns Tested

Tests are run for five minutes, with each traffic source generating one packet per second according to the below traffic patterns.

1) One-to-Many: The one-to-many traffic patterns test routing protocols in a typical server/client fashion. In a one-to-many test there is a single source node and many receiver nodes. 25% of the devices are designated as group members interested in receiving data from the source node. The source node then sends broadcast traffic to each of the receivers.

This traffic pattern is similar to classic multicast patterns, which are often seen in military tactical networks [5]. Voice traffic and situational awareness are typically transmitted via a one-to-many traffic pattern.

2) Many-to-One: In the many-to-one traffic pattern we run the same experiment as above, but each receive node generates and sends a message back to the source. This traffic pattern tests the network under much heavier conditions since each group member now is a source for packets.

This traffic pattern models a case in which sensors are distributed throughout a network with a single processing unit that wants to aggregate data from the sensors. The processing unit sends commands to the sensors and the sensors send back what data they have collected.

IV. Link Quality Measurements

To evaluate the connectivity between devices, we wrote a lightweight program called Beaconer to collect link statistics. Beaconer broadcasts a 64 byte packet one hop every 3 seconds and records packet receptions of neighbors. By comparing the number of received Beaconer packets between userswith the

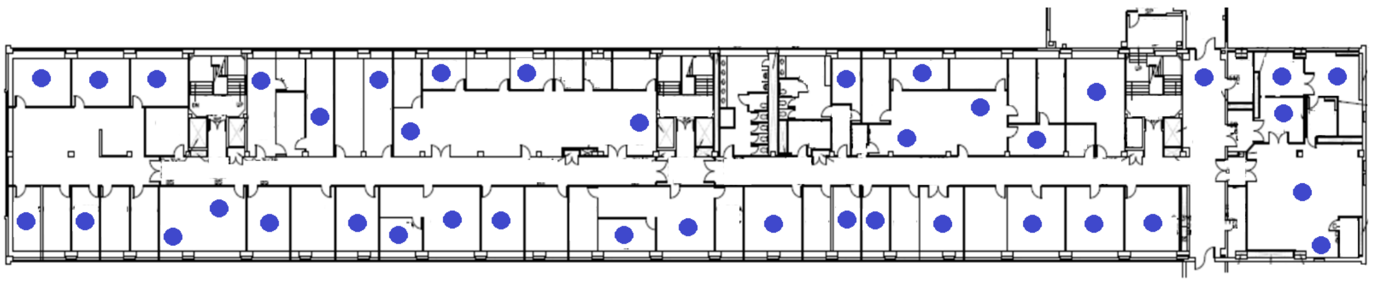

Fig. 1: Sample office topology. Dots represent Android phones.

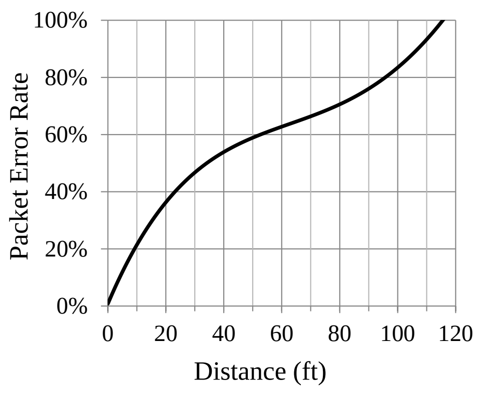

Fig. 2: Observed packet error rate vs. distance between users

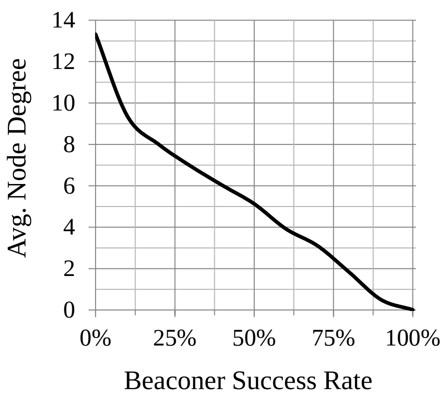

Fig. 3: Average node degree of sub-graphs with links of a minimum Beaconer success rate

number of transmitted Beaconer packets, we get estimates of packet error rates for particular links between users throughout experiments. Using this data and the locations of the phones, we created a Packet Error Rate (PER) vs. Distance graph for the different topologies. The average curve for data collected over all tests is shown in Figure 2.

Using data collected from Beaconer, we can more closely evaluate channel characteristics. While some phones might have up to a dozen other phones within 30 feet, in reality, only few of those links are reliable. If we consider sub-graphs of the network that have only links of some minimum Beaconer success rate, we get a better picture of the reliable network topology. Figure 3 is a plot of the average node degree when links are limited to a maximum packet error rate (minimum Beaconer success rate) for a sample 60 node test. The 0% Beaconer success rate represents users that exchanged at least one Beaconer message, but not sufficient numbers to register a full percentage point. Users that exchange at least one message have 13 neighbors on average. With 50% Beaconer success rate, a node only has an average of 5 neighbors. For a 90% Beaconer success rate, a node can expect to have at most 1 neighbor, meaning that there are very few reliable links throughout the network.

Continuing with the sub-graph approach where only links that have a minimum Beaconer success rate are included, we consider the number of users that actually reach one another and the average path length between them. For the 0% Beaconer success rate, a path exists between all users, and the average distance between users is 3.6 hops. For 50% Beaconer success rate, only 34% of users are connected between one another, and the users that remain connected have an average distance of 4.4 hops between themselves. For 70% Beaconer success rate, a meager 9% of users are connected. As shown by our tests, reliable paths between users unfortunately cannot be expected to exist.

These tests were conducted in an office with presumably negligible external interference and no mobility. Even in such a benign environment, the over-the-air links between nodes are very lossy. This demonstrates the challenges that the Internet of Things faces as it tries to connect 100's of nodes. A tactical network would see connectivity potentially even worse than this. The channel would be contested by other users and jammers, and we expect the nodes to be mobile.

V. Test Results

To measure performance, we look at the percentage of messages successfully delivered, as well as the total network load of packets transmitted over-the-air. While some protocols deliver a high percentage of packets, they do so at such a high cost that may not be supportable in heavily resource constrained environments.

We test two resiliency levels for GCN: high and low. High resiliency sets the relay parameter R to 6, and low resilience sets R to 3. We note that these resiliencies exist on a continuum so they could be changed to meet the needs of a given network.

We examine results of 30, 60, and 90 node tests. We vary the number of nodes to increase network density, not network diameter.

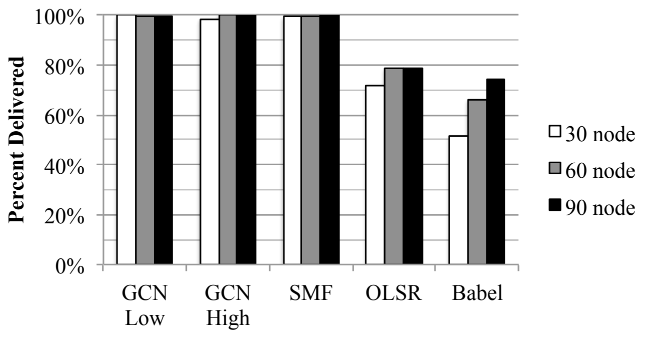

Fig. 4: Percentage of Packets Delivered, One-to-Many

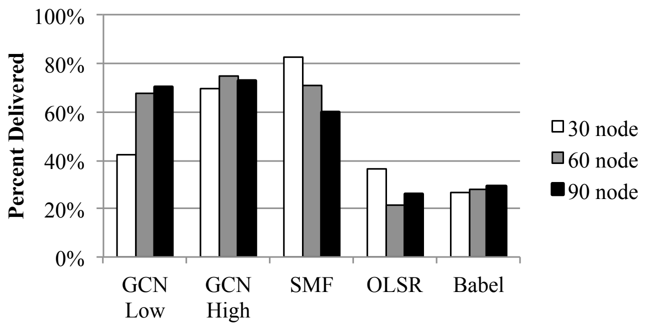

Fig. 5: Percentage of Packets Delivered, Many-to-One

A. Delivery

1) One-to-Many: In the one-to-many traffic pattern a single source node is sending data to many receive nodes. One quarter of the devices are designated as receive nodes.

We have found the over-the-air results follow what we saw in NS3 simulation with GCN outperforming the other routing protocols. Figure 4 shows the percentage of packets delivered for the different routing protocols and parameters run. GCN at both low and high resiliencies was able to deliver close to 100% of the packets. Babel only managed to deliver half of the traffic and OLSR delivered between 70% and 80%. Like GCN, SMF delivered close to 100% of the packets. Though SMF delivered packets successfully, we will see in later sections that this success came at huge cost with respect to network resources utilized. We can attribute GCN's good performance to successful group discovery and tunable resiliency: the nodes were able to construct the group and exchange data successfully. OLSR and Babel often failed to establish routes between participating devices and thus could not send data.

2) Many-to-One: The many-to-one traffic pattern is more demanding on all of the protocols. Now a quarter of the devices are active senders and they each want to deliver traffic to the same receive node. No protocol surveyed was able to deliver 100% of the required traffic.

As seen in Figure 5, SMF and GCN on high resiliency have the highest delivery rates. Interestingly, as network density increases, GCN maintains performance while SMF's performance declines. At only 30 nodes SMF delivers over 80% of packets while GCN at high resiliency delivers 70%. As we move to 60 and 90 nodes, we see SMF's delivery rate decline to 60% and GCN's high resiliency performance staying the same.

We observe that GCN on low resiliency performs better as network density increases. In a 30 node network, low resiliency GCN delivered only around 40% of the packets, but in a 90 node network, both low and high resiliency GCN performed similarly, delivering 70% of the packets. This change is caused by the difference in node density. Under high density, there are many good links to choose from, resulting in high delivery, but in a sparse network, there may be only one good relay to choose. Low resiliency GCN has a low probability of choosing this good relay, whereas the increased number of relays selected by high resiliency GCN results in a much greater chance of selecting this good choice. The consequence of this higher likelihood of selecting the best relay is that high resiliency GCN uses more network resources.

In this Many-to-One scenario, OLSR and Babel both could only deliver approximately 30% of packets to their intended destinations. The reason for this is because each of these protocols have explicit links that a packet must traverse, and often times, those links are lossy or unavailable. This causes a large number of packets to not be delivered.

To explore why SMF's performance declined as density increased we must look at the network resources used. In the next section we will see that SMF uses far more resources than other protocols and causes congestion in the network, lowering its delivery rate.

B. Network Resources Used

To measure the total resources used in a scenario we recorded how many bytes were transmitted-over-the air for each protocol. We include all data or control packet sent by all nodes across the network. For the wireless multi-hop ad-hoc networks of interest, such as tactical networks or IoT, link rates are typically low, so transmitting as little over-the-air is essential. Fewer resources used is better.

1) One-to-Many: Figure 6 shows the amount of resources used in megabytes (MB) in the One-to-Many traffic pattern. From simulation we would expect SMF to use the most data in this traffic pattern. Interestingly, Babel uses the most resources. This is because Babel transmits an enormous amount of control information as it constantly tries to repair seemingly broken paths.

For the 30 node topology GCN with low resiliency transmits only 1 megabyte of data, and GCN with high resiliency (higher redundancy) transmits about 4 MB. SMF uses over 5 MB, more than GCN on high resiliency and much more than GCN on low resiliency. All three protocols delivered 100% of the traffic in this scenario, so GCN with low resiliency is optimal for this topology.

As we scale to 60 and then 90 nodes we see little change in data usage by GCN on low resiliency, while GCN on higher resiliency does use more data as density increases. The reason for this is because when there were only 30 nodes in the network, GCN on high resiliency wishes to activate many relays, but there were not enough to nodes to actually become

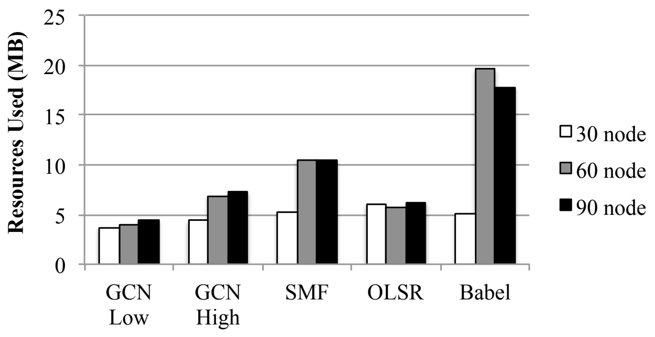

Fig. 6: Network resources used, One-to-Many

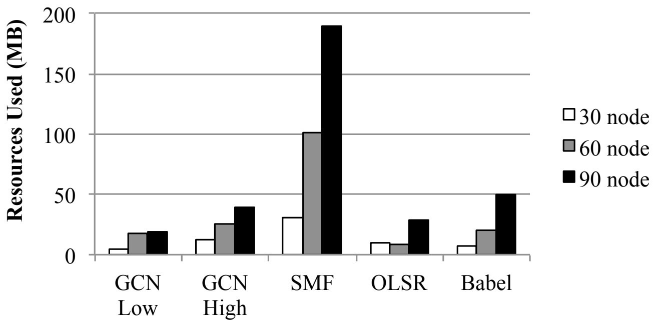

Fig. 7: Network resources used, Many-to-One

relays. If a node only has three neighbors, no matter how high the resiliency parameter that node can have at most three neighboring relays. Increasing the topology to 90 nodes allows GCN's high resiliency to find the relays it wants, so it sends more traffic through the network.

2) Many-to-One: Figure 7 shows the amount of resources used in bytes in the Many-to-One traffic pattern. This pattern requires many nodes to send data back to a single node, so there are far more flows in the network than there were in the One-to-Many pattern. Due to the increased number of packets, all of the protocols transmit more over-the-air. At any network density SMF, uses more resources than any other protocol. Its high delivery rate clearly comes at a cost to the network. Networks with low data rates, like military tactical networks or IoT, would be unable to handle the sheer volume of data transmitted over-the-air.

Of course, all of the protocols are going to use more data as the number of nodes increases. Since one quarter of the nodes in the network are designated as sources, the 90 node test has far more sources than the 30 node network does. However, GCN scales well with the more intensive traffic pattern, while SMF does not. GCN's data usage increases nearly linearly: GCN uses 10MB, 25MB, and 40MB at 30, 60, and 90 nodes respectively. Since SMF is a flooding approach, SMF's usage skyrockets from 30MB to 100MB to 190MB.

For the 90 node topology GCN with high resiliency uses 40MB, and GCN with low resiliency uses 20MB. OLSR and Babel transmit 25 and 50 MB, respectively. GCN uses a similar amount of resources as Babel and OLSR in the Many-to-One scenario, but it achieves a much better delivery percentage in doing so. Recall from Figure 5 OLSR and Babel struggled to deliver 30% of the data, while GCN delivered 70%. As can be seen, OLSR and Babel would a poor choice for a military network or IoT because the links between users are very lossy. GCN, however, is more likely to deliver data and tolerate the lossy links.

VI. Conclusion

In this paper, we performed the first over-the-air testing of Group Centric Networking on a large scale Android phone test bed. These results validate previous results from simulation that confirm GCN performs well in tactical environments.

Military tactical networks and IoT require efficient routing protocols because they need to operate in highly contested environments. As shown in our tests, traditional routing schemes fail to meet these networks' needs: flooding offers high delivery rate and high overhead or and routing protocols offer low delivery rate and low overhead. GCN, however, provides the best of both worlds with a high delivery rate and low overhead.

GCN is an excellent step towards tactical networking. Next steps for GCN includes increasing end-to-end reliability by developing a GCN specific transport layer protocol.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] Z. Sheng, S. Yang, Y. Yu, A. Vasilakos, J. Mccann, and K. Leung, "A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities," *Wireless Communications, IEEE*, vol. 20, no. 6, pp. 91–98, 2013.

[3] T. Watteyne, A. Molinaro, M. G. Richichi, and M. Dohler, "From manet to ietf roll standardization: A paradigm shift in wsn routing protocols," *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 4, pp. 688–707, 2011.

[4] G. Kuperman, J. Sun, B.-N. Cheng, P. Deutsch, and A. Narula-Tam, "Group centric networking: A new approach for wireless multi-hop networking to enable the internet of things," *arXiv preprint arXiv:1511.08114*, 2015.

[5] C. Wilson, "Network centric operations: Background and oversight issues for congress." Defense Technical Information Center (DTIC) Document, 2007.

[6] R. Ramanathan, R. Allan, P. Basu, J. Feinberg, G. Jakllari, V. Kawadia, S. Loos, J. Redi, C. Santivanez, and J. Freebersyser, "Scalability of mobile ad hoc networks: Theory vs practice," in *MILCOM 2010*. IEEE, 2010, pp. 493–498.

[7] P. Jacquet, "Optimized link state routing protocol (olsr)," *IETF RFC 3626*, 2003.

[8] J. Chroboczek, "The babel routing protocol," *IETF RFC 6126*, 2011.

[9] J. Macker, "Simplified multicast forwarding," *IETF RFC 6621*, 2012.

[10] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA'99. Second IEEE Workshop on.* IEEE, 1999, pp. 90–100.

[11] R. Khanduri, S. Rattan, and A. Uniyal, "Understanding the features of ieee 802.11g in high data rate wireless lans," *International Journal of Computer Applications*, vol. 64, no. 8, 2013.

[12] S. Avallone, A. Pescape, and G. Ventre, "Analysis and experimentation of internet traffic generator," *proc. of NEW2AN*, pp. 70–75, 2004.