Simulation-Assisted Routing Protocol Design (SARPD)

Dr. Mark Stell
Dr. Cam Tran
Dr. Albert K. Legaspi
SPAWARSYSCEN San Diego
53560 Hull Street
San Diego, CA 92152
Mark.Stell@navy.mil, Cam.Tran@navy.mil, Albert.Legaspi@navy.mil

Keywords: Simulation-Assisted Design, Routing Protocols, JTRS Radio

ABSTRACT: *A case study in the use of a simulation-assisted routing protocol design environment (SARPD) during the development of military Line-Of-Sight (LOS)/Beyond-Line-of-Sight (BLOS) radios is presented. It focuses on routing protocols for the JTRS-surrogate VRC-99A/B radio. Modern SARPD environments can be an extremely cost-effective development option, when compared to field trials.*

1. Introduction

The routing protocols used on large-scale IP data networks are examples of massively distributed real-time algorithms. Subtle algorithm behaviors can have dramatic effects on network performance. Therefore, it can be highly cost effective to use a simulation-assisted routing protocol design environment (SARPD, pronounced "sarp-dee"), rather than immediately going to field trials. The development of military networking devices is an especially attractive candidate for applying these environments.

The Simulation Modeling and Analysis for Naval Battleforce Networks (NBN) effort supports communications technology developers at Space and Naval Warfare Systems Center San Diego (SSC San Diego). The NBN program goals include better adaptive, mobile, wireless networks connecting multiple Naval platforms within a battle group, as well as joint battlefield scenarios. In this case, modeling supports integration of Joint Tactical Radio System (JTRS)-like radios, such as the VRC-99A/B, in Naval data networks.

An interesting problem, namely the subnet-relay problem, occurs in line-of-sight and beyond-line-of-sight (LOS/BLOS) networking radios. The IP Subnet model requires that the link layer allow any IP device to send a packet directly to any other IP device on that subnet [1]. However, limited radio range, combined with mobility of nodes, means wireless networks do not always comply with the IP Subnet model. In fact, LOS/BLOS radios often require dynamic relaying between nodes as the need for, and location of, the relays can change frequently. Thus, the subnet relay problem is central to integrating LOS/BLOS radios into larger IP data networks.

Various solutions have been tested. Originally, physical implementations were developed and tested in field trials. However, these efforts were time consuming and expensive. In an effort to reduce risk and cost, a number of efforts are under way to use Simulation-Assisted Design in order to test the performance of routing protocols, before going to field trials of proposed subnet relay solutions.

Open Shortest Path First (OSPF) Point-to-Multipoint mode can solve the subnet-relay problem, but a previous simulation study [2] found this mode to be poorly suited to typical military LOS/BLOS networks.

The same study [2] also reported a simulation-assisted design effort. The authors modeled a proposed extension to the OSPF routing protocol. The extension was based on the Mobile Ad Hoc Network (MANET) protocols. In general, they found the design to be both efficient and well suited to LOS/BLOS networks. They are currently implementing this design for near-term field trials. Simulation-assisted design provided much higher confidence that the investment in implementing and testing the design is justified.

We are using simulation-assisted routing protocol design to examine a different solution to the subnet-relay problem. This design requires the radios to emulate the IP subnet model, based on dynamic link-layer bridging. However, concerns about the speed of convergence of the protocol, as well as its bandwidth consumption, lead us to test the design as a simulation model before making any final implementation recommendations.

Existing high fidelity products to model the physical, link, and network layers provided a cost-effective platform on which to rapidly build the test environment.

2. Protocol Design

While roughly based on the VRC-99 radio protocol, we chose to model a similar, but more general approach that can be applied to any LOS radio.

The VRC-99 radio uses a version of the Layer 2 IP Learning Bridge to enforce the IP subnet model. It has two independent parts: a wireless part and an IP part. The wireless part discovers the link state of the radio network and uses that information for wireless routing. It uses radio numbers, not IP addresses, in the link state information.

The IP part of the VRC-99 learning bridge protocol records the source Ethernet addresses of packets received from other radios, and uses that information to identify the destination radio for packets that are to be sent over the air. When there is no information about a packet's destination, the packet is sent to all radios.

The design used in this simulation was slightly different than the VRC-99 protocol. It used Layer 2 routing based on Ethernet addresses[1]. The proposed protocol is based on the experiences of the Navy's Automated Digital Network System (ADNS) and the Navy's VRC-99 radio user community.

The potential advantages of either Layer 2 bridging or routing are 1) they are easy to implement, and 2) the resulting wireless network integrates well into larger IP internetworks, e.g., Navy-wide or DoD-wide IP networks.

The target environment consists of LOS/BLOS radios that experience minimal propagation delays (on the order of time-of-flight of 120 miles, or less), and that have bandwidths on the order of 100,000 bits per second per radio, or higher [3]. Our proposed protocol is designed for small numbers of radios per subnet, ideally 10-16 radios per subnet. The protocol assumes each radio is directly connected to a router that could be a commercial-off-the-shelf (COTS) product, or a workstation running routing software (see Figure 1). Finally, the design assumes a time-varying topology such that sometimes a given radio is a single hop away and sometimes it is more than one hop away.

The technical goals of the design include:

- rapid response to changing topology caused by node mobility and/or link failure (rapid being less than one minute),
- straightforward integration into general purpose IP data networks, e.g., ADNS,
- low routing protocol overhead,
- separation of the router and radio to maximize COTS and minimize development,
- ability to use many physical layers, e.g., CSMA, TDMA, FDMA,
- ability to interoperate with multiple router vendors, i.e., no requirement for special vendor support
- ability to support multiple routing protocols, e.g., OSPF, EIGRP, IS-IS
- extreme simplicity for ease of implementation

The general architecture is illustrated in Figure 1. Within the wireless network routes are based on Ethernet addresses. The radio will always see a router Ethernet address as the source and the Ethernet address of another subnet router, or a broadcast/multicast address, as the destination. This invariance is the basis of this routing proposal[2]. This allows the networking radios to forward packets between a very small set of Ethernet addresses
.

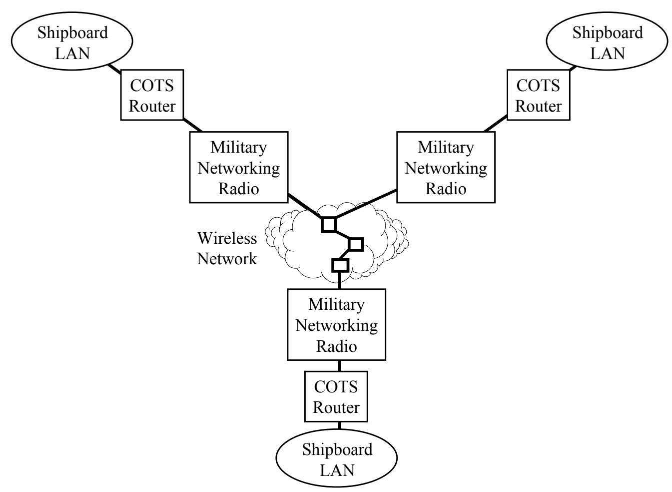

Figure 1: Physical Relationship of Routers and Networking Radios

In this architecture, the routers are configured as if they were on an ordinary Ethernet subnet. It is the responsibility of the networking radios to ensure that the IP subnet model holds for the wireless network, at least

[1] We used a local ID to represent the Ethernet address.

[2] Layer 3 tunneling across the wireless network causes the same invariance.

as far as Layer 3 devices, i.e., the routers, are concerned. This concept is illustrated in Figure 2.

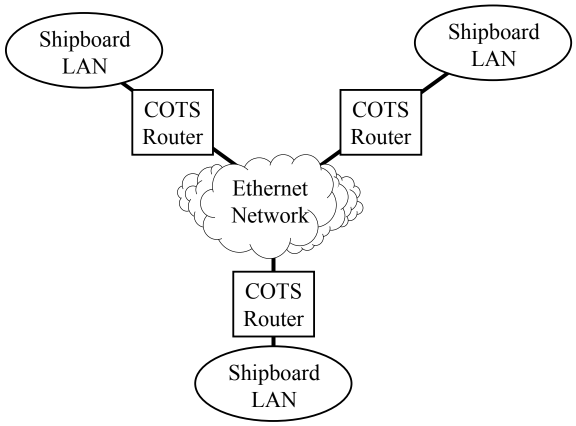

Figure 2: Logical Relationship of Routers

The appearance of a standard IP subnet allows virtually any COTS router to be used with the networking radios. It also allows the routers to run any routing protocol over the networking radios.

In our proposed protocol, the multicast addresses are translated into broadcast addresses, and the networking radios ensure that all routers in the subnet receive the broadcast/multicast.

It is possible for the networking radios to learn the Ethernet address of their directly attached router by monitoring the source address of all Ethernet packets, e.g., ARP requests and routing protocol packets.

The wireless routing protocol can then distribute the router Ethernet addresses throughout the network. The networking radios use a simple Proxy-ARP type protocol to respond to router ARP requests. Neither the ARP monitoring nor the Proxy-ARP responses were included in this simulation, but these techniques have been used successfully in previous Naval wireless networking implementations.

3. Description of the Protocol

The intention of this design is that the cost of each link be defined by the link layer (Layer 2). In the case of the VRC-99, the link layer continuous adjusts the link bandwidth to each neighboring radio based on receipt of link connectivity packets. The link layer then assigns a routing protocol cost. In the current simulation, the link costs were manually assigned[3].

Each radio used periodic, unreliable, broadcast of topology data to distribute the link-state data. Each radio's link state broadcast incorporated all link state data received from other radios, as well as its own information. In the case of the VRC-99, the link state data included radio identifiers which are 4 bits long. In the case of this simulation, the link state data consisted of 48 bit router Ethernet addresses[4], a node connectivity matrix containing link costs, and sequence numbers for each node's entries in the connectivity matrix. Various update periods were used, with the highest rate being once per second.

Similar to OSPF, the Dijkstra Shortest-Path algorithm was used to compute unicast routes [4]. The Dijkstra/Prim Minimum Spanning Tree algorithm, rooted at the multicast source, was used to find multicast routes [5].

If a networking radio did not receive routing update packets from an adjacent (one-hop-away) radio for a predetermined time, the routes to that neighbor were removed, and new routes (if any) were calculated

4. Methods

All simulations ran for 300 seconds.

The QualNet modeling and simulation tool served as the underlying platform for building the SARPD environment. The major functionality consisted of standard QualNet functions, including the 2.4 GHz 802.11b channel, MAC and IP models. Omni-directional antennas were used with a path loss model that considered ground reflection on flat earth (Two-Ray).

To simplify the development of the model, each node was both a traffic generator and a wireless radio/router. Networks outside of the wireless subnet were modeled as traffic sources and sinks.

4.1 Speed of Convergence

The purpose of the routing protocol is to facilitate delivery of end-user (application) data. Therefore, application packets were generated at a known time, and

[3] A detailed link layer model is being developed. It will include bundling of multiple packets in order to better use available TDMA slots, similar to the VRC-99.

[4] This simulation used pseudo-Ethernet addresses, since our simulator did not have actual addresses.

the reception of each unicast and multicast packet, at every node, was recorded. Based on radio range calculations, it was possible, in some scenarios, to calculate the time difference between a node coming into radio range and the time the first application packet was received by that node. This time difference was used as the definition of routing protocol convergence.

The effect of mobility on the speed of convergence was a central concern. It was examined in several different scenarios.

1) In the least challenging mobility scenario, every node could contact every other node directly. Even in this case, however, the unicast forwarding table needed to be populated by the routing protocol, or packets would be discarded by the radio due to lack of a route.

2) In an intermediate mobility scenario, two small groups (4 nodes each) were fully connected internally, but the groups could not communicate. In this case, a mobile node moved into a position between the two groups, simulating an unmanned aerial vehicle (UAV).

3) In the most challenging mobility scenario, the nodes were initially fully connected, but during the simulation they moved to form a single threaded chain. For these tests, the traffic source was the head of the chain. This meant that traffic had to be relayed from the source through every other node to reach the final node in the chain. This required the routing protocol to make dramatic changes to each node's forwarding table.

The application traffic sources were constant bit rate traffic generators. For multicast traffic every node joined the 235.0.0.1 multicast group and node one was always the multicast source. A 512 byte multicast packet was transmitted from node one to all the other nodes on even numbered seconds.

For unicast traffic, the highest numbered node (usually node 10) was the source. This node would send a 512 byte unicast packet to all of the other nodes on odd numbered seconds. In addition, each node would send a 512 byte unicast packet back to the highest numbered source on odd numbered seconds.

4.2 Protocol Bandwidth Consumption

To measure protocol bandwidth consumption, the amount of bandwidth used by the Layer 2 (subnet-relay) routing protocol was measured. No attempt was made to estimate the additional overhead any Layer 3 routing protocol, e.g., OSPF, would add.

In the case of the VRC-99, the link connectivity packet (which communicates link state information) is 144 bytes long, regardless of the number of radios in the network. It provides the link state of the entire network.

In the case of this simulation, the protocol sent the entire connectivity matrix inside every routing protocol update and the size of the matrix varied depending on the number of nodes in the subnet. For this simulation, between 4 and 16 nodes were used. In addition, the frequency of routing protocol updates was varied between one per second and one every ten seconds.

5. Results

Development of the unicast routing protocol occurred much as expected. However, a number of interesting observations occurred when modeling the multicast routing protocol in a wireless environment.

The initial design produced broadcast storms in situations whenever broadcast or multicast packets were relayed. This design populated a multicast forwarding table based on the Dijkstra/Prim minimum spanning tree, where the tree was rooted at the multicast source. For a radio to determine if it needed to relay traffic from a given source, it needed only determine if it was a non-leaf node on the tree.

However, this simple approach ignored the fact that, when there were several relay nodes in a row, nodes upstream in the chain (closer to the source) would receive the multicast packet a second time when downstream nodes relayed it further along the chain.

When the upstream relay received the packet the second time (from a downstream relay), the upstream node forwarded the packet back to the relay it had just received it from. This meant every multicast packet passed endlessly back and forth between relay nodes, whenever there was more than one relay node in the network.

In the case of the VRC-99, this is not a problem. The radio that sends a packet specifies (in the packet header) the radios that are to process the packet. When a packet needs to be relayed to multiple destinations, the source radio selects the relaying radios. The packet header specifies which radios are to relay the packet to which destinations.

For the VRC-99, the relaying radio may need to use further relays. It does this by repeating the process performed by the source radio.

Additional solutions to this problem were explored in this simulation. One solution involved keeping a cache of recently forwarded packets. Another solution involved noting the link layer address of the node that transmitted the packet, and only forwarding packets that came from the correct neighbor. For this simulation, in order to better separate the link and physical layers, a simple cache was implemented.

The second observation was that multicast applications on wireless networks often received a multicast packet several times, if multicast relays were used. For example, nodes near the source often received the multicast when it was originally transmitted by the source. Later, when the multicast was retransmitted by the relay, many of the same nodes received the packet a second time.

In the case of the VRC-99, no cache is needed. The packet header specifies which radios are to receive the packet. Therefore, all needed information is in the header. However, when using an acknowledged protocol, a sliding window is maintained (using sequence numbers) to generate Acks and requests for retransmission. There is a separate sliding window for each radio-to-radio link.

For purposes of this simulation, it was assumed that multicast applications could tolerate multiple receptions of the same packet.

5.1 Speed of Convergence

A particular concern was convergence time in different mobility scenarios. Each mobility test was given a unique identifier.

Test 1A: No mobility, nodes fully isolated. This test provided baseline data. All the nodes were more distant that the maximum radio range. In this case, routing protocol packets could not be exchanged. The broadcast and multicast packets were transmitted, but the unicast packets were discarded due to the lack of a route. Neither unicast nor multicast packets were received by any node.

Test 1B: No mobility, nodes fully connected. All nodes were within radio range of all other nodes. Routing protocol packets were exchanged between all nodes during the first update interval (at time = one second). Shortly after the first routing protocol update, the first unicast application packet was sent. All nodes received the unicast packet during the first second. Thus, the unicast routing protocol had converged by the time the first application packet was sent (time = one second).

Multicast application data were first transmitted at time = two seconds, and every node received the multicast at this time. Therefore the multicast routing protocol also converged by time = two seconds.

Test 1C: Simple mobility. This was the least challenging mobility test. This test verified the speed of convergence after a connection had been lost and re-established.

During the first 100 seconds, all nodes were within radio range of all other nodes. As in Test 1B, the first unicast and multicast packets were successfully received by all nodes. Unicast application packets were consistently received up to time = 99. Multicast application packets were consistently received up to time = 98.

At time 100, all nodes moved out of radio range of all other nodes (fully isolated). The movement was instantaneous, i.e., the nodes moved from within radio range to outside of radio range in a single movement. Between 14 and 17 seconds after the movement, the routing protocol formally declared all neighbors lost and all routes invalid.

At time 200, all nodes moved back to within radio range of all other nodes (fully re-connected). The first unicast application packet was received at time = 201. The first multicast packet was sent at t = 202 and was received during the same second. Thus, the time to re-converge (multicast and unicast) was again on the order of one update interval (one second).

Test 1D: This was the most challenging test for the routing protocol. All the nodes started in close proximity, able to send directly to every other node (fully connected). The nodes then moved to form a chain. In this chain, every node, except the first and last nodes, was connected to two other nodes (an upstream neighbor and a downstream neighbor). For unicast to go from the source (node 10) to the other end of the chain (node 1), it had to be relayed by every other node in turn, i.e., it had to be relayed by nodes 9, 8, 7, 6, 5, 4, 3, and 2, in that order. This required a complete reorganization of each node's forwarding table.

Multicast, which originated at node 1, also had to be relayed across all the other nodes to reach the far end of the chain (node 10, in this test). Thus, each node's multicast forwarding table also had to be reorganized.

The movement used to transition from fully connected to a strict chain was programmed to occur smoothly, in a series of small steps. This meant that, as the nodes transitioned from fully connected to a chain, the selection of relay nodes changed continually. Node 10 experienced the maximum speed of any node, it moved at 8.5 m/s and the radio range was 440 meters. This meant this node exceeded the radio range in approximately 52 seconds.

The first observation was that, the farther downstream a node was, the longer the interruption of traffic reception. This was because failure of any upstream relay disrupted traffic for all downstream nodes.

Unicast packets were continually received by nodes 10 (the source) and 9. Node 8 did not receive traffic for 40 seconds. Traffic for the other nodes was disrupted for 102 seconds, from time = 99 to time = 201. All nodes were again receiving traffic, beginning at time = 201, continuing until the end of the simulation.

Multicast packets were continually received by node 2. Node 3 did not receive traffic for 24 seconds. The other nodes had no reception for 102 seconds, from time 98 to time 200. All nodes were again receiving traffic beginning at time = 200 until the end of the simulation.

Thus, there was a long period, about 100 seconds, when most nodes were not receiving traffic. However, as the structure of the chain stabilized, at around time 200, the routing protocol re-converged and application traffic was again delivered end to end, even through it meant that the data had to be relayed up to 8 times.

Thus, the convergence time for this mobility scenario is about 100 seconds. This exceeded our pre-defined convergence time limit of 60 seconds.

Test 1E: This test began with all nodes fully connected, then the nodes moved according to the built-in QualNet random way-point algorithm.

The best summary of this data is that connectivity of individual nodes was repeatedly established and then lost. No attempt was made to determine when the nodes were, or were not, in radio range.

Test 1F: This test simulated a mobile relay, such as an unmanned aerial vehicle (UAV), connecting two groups of nodes. This test was of intermediate difficulty for the routing protocol.

The first group consisted of nodes 1, 2, 3, and 4. These nodes were fully interconnected. The second group consisted of nodes 6, 7, 8, and 9. These nodes were also fully interconnected. However, there was no communications between groups, i.e., the groups were out of radio range of each other.

Node 5 was initially out of radio range of all nodes. As the test progressed, node five moved to a point equidistant between groups 1 and 2, and within radio range of both groups.

Figure 3 shows the layout for test 1F after the relay node is in position. Group one is shown on the left side of the canvas. Group two is shown on the right side of the canvas. The relay node, node 5, is between and above the two groups. The circles seen around node 5 are an

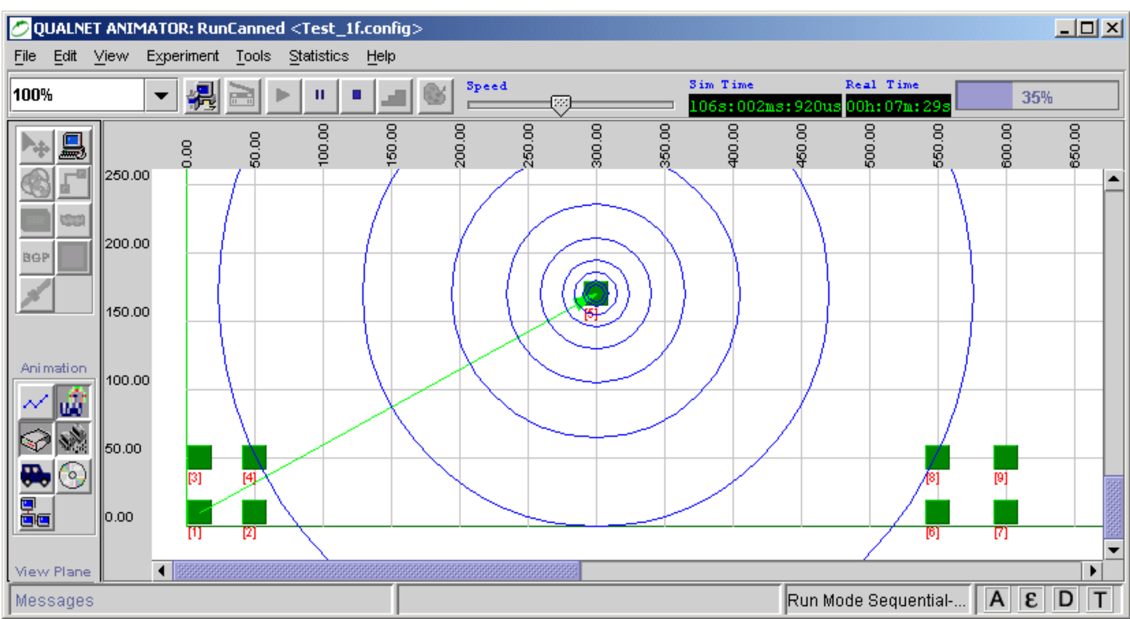

Figure 3: Test 1F Topology

animation of a radio broadcast. The arrow between node 1 (the multicast source) and node 5 shows data being sent to the relay node.

Unicast results: The relay node, node 5, initially began receiving unicast application packets from node 9 at time = 55. Node 4 was the first member of the distant group to receive a unicast packet, at time = 57. The rest of the nodes received the first packet at time = 69. Therefore, the time for the unicast protocol to converge was 14 seconds (69 - 55).

Multicast results: The multicast results are presented in Figure 4. The relay node first received a multicast packet at time = 56. All of the nodes in the distant group also received the first packet at time = 56. Thus, the multicast protocol was effectively converged in one update interval. Note, however, that a decision to relay the multicast packet to any node in the distant group effectively meant relaying the packet to all the nodes in that group, since all the nodes in the group were within radio range of the relay node.

5.2 Protocol Bandwidth Consumption

This section focuses on the efficiency of the proposed protocol. In particular, it looks at the amount of bandwidth the protocol used.

The proposed protocol broadcasts the entire link connectivity matrix every "update-interval" seconds. The broadcast is unreliable. Nodes that receive the broadcast use it in their routing calculations. Nodes that do not receive it cannot ask for it to be repeated.

The amount of bandwidth used by the protocol is not affected by user traffic, in contrast to "on-demand" wireless routing protocols. Bandwidth consumption is not affected by events in other parts of the subnet, such as link failure or link restoral, in contrast to link-state routing protocols. Thus, bandwidth consumption is very predictable for any give configuration.

<u>Test 2</u>: In this test the size of the link connectivity matrix was varied. The update rate remained constant at 1 routing protocol packet per second per node. Nodes did not move during this test.

As seen in Figure 5, bandwidth utilization increased as the number of nodes in the connectivity matrix increased. Four nodes consumed about 500 bits per second per node. Sixteen nodes consumed about 5,000 bits per second per node.

The rate of increase in bandwidth consumption was on the order of N^2, where N is the number of nodes in the connectivity matrix. Bandwidth consumption increases rapidly because the connectivity matrix is two dimensional and is contained in all routing protocol packets.

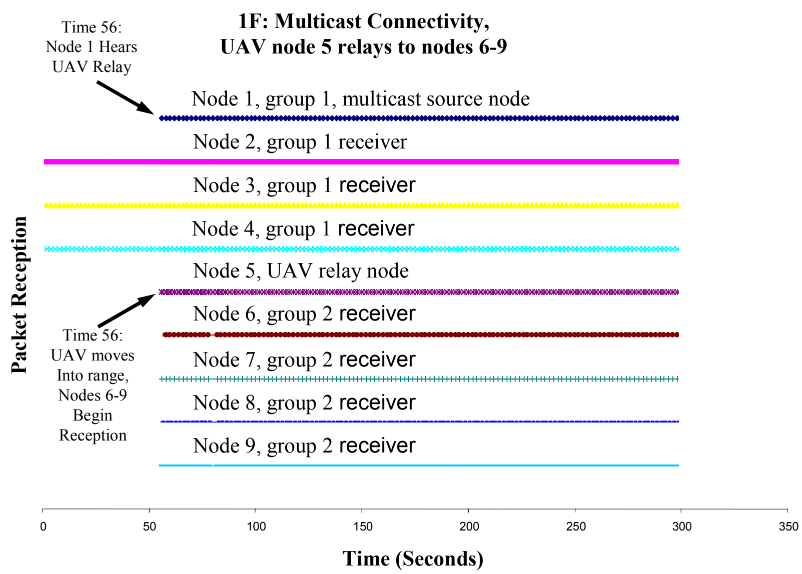

Figure 4: Multicast Connectivity, Mobile Relay Node

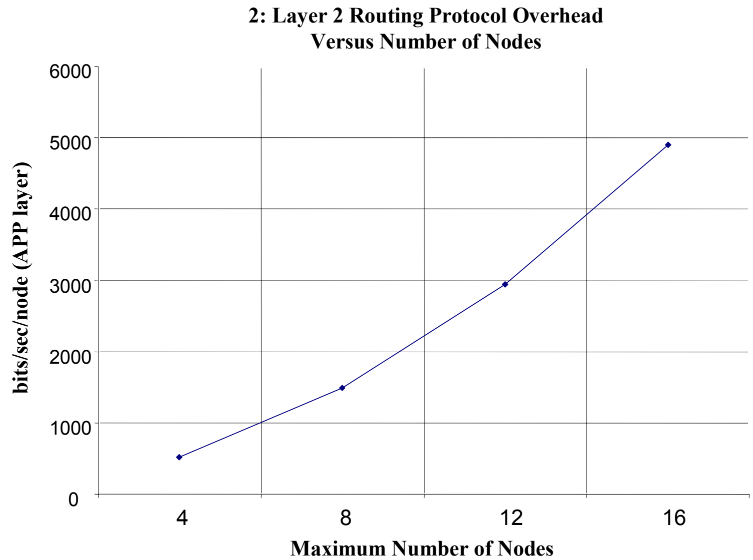

Figure 5: Protocol Overhead Versus Number of Nodes

Note that the numbers shown in Figure 5 are routing protocol overhead at the application layer, before the addition of UDP, IP, link, security, or physical layer headers. To some extent, the degree to which the size of the routing protocol packets will grow, as they pass through the other layers, is dependent on how the protocol is implemented and the details of the specific networking radio.

Note also that Figure 5 does not include any routing protocol overhead introduced by the routing protocol running on the larger IP internetwork. For example, if this subnet is embedded in a system that runs OSPF, then additional routing protocol overhead will be introduced by OSPF packets as they are transmitted across the subnet. The amount of overhead introduced by the overarching routing protocol depends on what routing protocol is being used and the specific configuration of that protocol on the wireless subnet.

Test 3: This test examined the amount of bandwidth consumed when the update interval was varied. The results are shown in Figure 6. The connectivity matrix was configured for 16 nodes. Update intervals of 1, 2, 4, 6, 8, and 10 seconds were tested. As seen in the figure, bandwidth consumption increased as the frequency of updates increased.

When the update interval was one update every 10 seconds, then the protocol consumed about 500 bits per second per node. When the update interval was one update per second, then the protocol consumed about 5,000 bits per second per node.

The advantage of longer update intervals is reduced bandwidth consumption. The disadvantage of larger update intervals is slower convergence after nodes move. The mobility tests, Tests 1A-1F, used an update rate of 1 per second. The effect of longer update intervals on convergence time has not yet been examined.

The numbers in Figure 6 are once again overhead at the application layer, before UDP, IP, Link, security, or Physical layer headers have been added. And, they do not take into account the overarching routing protocol used in any larger IP network.

6. Summary

In general, this extremely simple protocol was able to facilitate the delivery of end-user (application) packets. It should be relatively simple to implement it in most military networking devices.

Also, the independence of the subnet relay routing algorithm from the routing protocol used in the larger internetwork allows the larger network to choose any routing protocol.

One observation was that multicast applications need to tolerate multiple receptions of the same packet, or some more advanced form of multicast packet filtering needs to be implemented, such as used in the VRC-99.

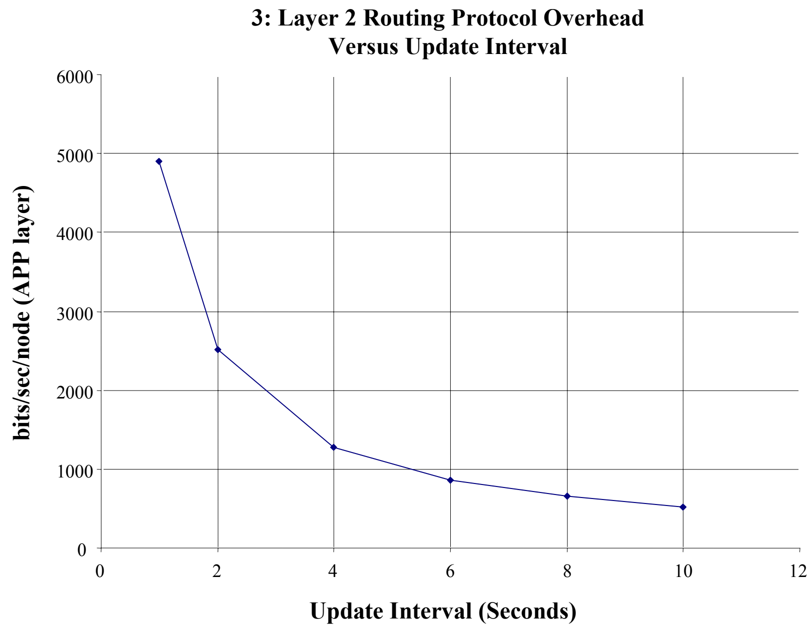

Figure 6: Protocol Overhead Versus Update Interval

6.1 Speed of Convergence

The ability of SARPD environments to produce quantitative estimates of convergence time significantly increased our understanding of the design.

In the least challenging mobility test, Test 1C, nodes moved from fully connected to fully isolated, then back to fully connected. In this test, the routing protocol converged in about one update interval, which was about one second.

In the intermediate mobility test, Test 1F, a mobile relay connected two stationary groups of nodes. In this test, the unicast routing protocol converged in about 14 seconds. The multicast routing protocol converged in about one update interval, i.e., one second.

In the most challenging mobility test, Test 1D, nodes moved from fully connected to a chain configuration. Throughout the test there was always at least one potential relay scheme that could have delivered the data to every node. However, during the test, many nodes did not receive any application traffic for about 100 seconds.

The reasons for the slow convergence in Test 1D are under investigation. The most likely cause is that, from time 100 to time 200, the pattern of relays was changing faster than the protocol could adapt. In particular, neighbors are not declared lost for ((update interval * number of nodes in the connectivity matrix) + 4), i.e., about 20 seconds. Only then would the protocol recalculate the routes. During the period of rapid change, the new pattern of relays may have been outdated by the time it was calculated.

When the structure of the chain stabilized, i.e., when the best path was clearly the chain, at time = 201, the protocol re-converged.

It is worth investigating whether or not TCP connections, such as those that underlie most e-mail, file transfer, and web transactions, would survive a 100 second outage. It is likely that the answer will depend on the particular TCP implementation, as well as the state of the TCP connection when the disconnection occurs. In general, we would expect most TCP connections to survive a 100 second loss of connectivity.

It is also worth investigating how often this particular mobility scenario occurs. In the other mobility scenarios the protocol converged more quickly.

6.2 Bandwidth Consumption

In the case of the VRC-99, the link connectivity packet (which communicates link sate information) is 144 bytes long regardless of the number of radios in the network. The 144 byte size includes the packet header and all but 12 bytes of slot overhead. This packet contains link layer information and is generated at the link layer. Therefore, there are no MAC/IP/UDP headers. The space reserved for an Ack packet is used for data if no Ack packet is actually needed.

In this simulation, we modeled an application layer routing protocol which was subject to MAC/IP/UDP headers.

In addition, bandwidth consumption was sensitive to the number of nodes in our connectivity matrix. For four nodes, at one update per second, the protocol consumed 500 bits per second per node. For sixteen nodes, the protocol consumed 5,000 bits per second per node.

Because the increase in the rate of bandwidth consumption was on the order of N^2, where N is the number of nodes in the connectivity matrix, this protocol is only appropriate for subnets with small numbers of nodes.

The bandwidth consumption was also sensitive to the update interval. With the largest number of nodes (sixteen) in the connectivity matrix, one update every 10 seconds consumed about 500 bits per second per node. One update per second consumed about 5,000 bits per second per node.

The bandwidth consumption results only considered application layer overhead. They did not include the overhead that would be added by other layers, such as the UDP, IP, link, and physical layers. For example, if we were to implement this protocol in the VRC-99 radio, we estimate that each routing protocol packet will increase by about 30%, before it is actually transmitted over-the-air. This assumes a routing protocol packet, at the application layer, of 644 bytes (true if the number of nodes in the connectivity matrix is 16). We assume the VRC-99 adds about 190 bytes to each application layer packet (20 bytes slot overhead, 32 bytes OTA packet header, 88 bytes reserved for an Ack packet, 22 bytes of IO processor and MAC header, 20 bytes for the IP header and 8 bytes for the UDP header).

No attempt was made to account for Layer 3 routing protocol overhead, such as that which would come from the overarching routing protocol run on the larger network. Total Layer 2 and Layer 3 routing protocol overhead depends to a large degree on which Layer 3 routing protocol is selected.

In general, networking radio designers should select an appropriate connectivity matrix size and update interval, depending on the radio characteristics and the target mobility environment. For our target environment, which consisted of about 10 radios and about 100,000 bits per second per node, the maximum settings are recommended, i.e., a connectivity matrix sized for sixteen nodes and an update interval of one per second. Other networking devices would likely lead to other choices.

6.3 Other Considerations

This approach could be implemented either at Layer 2 or at Layer 3, using tunneling. The disadvantage of the Layer 3 tunneling approach is that it would increase the overhead for each data packet. The additional overhead leads to a second problem; maximum size packets would have to be fragmented. The disadvantage of the Layer 2 approach is that it requires link layer packet forwarding. However, the advantages of the Layer 2 approach are likely to outweigh the disadvantages.

Because the bandwidth consumption of the proposed protocol grows on the order of N^2, the extensions to OSPF in [2] may be more attractive when there are more than 10-16 radios.

Future studies need to address quality of service (QoS). In many situations the networking radio will be the bottleneck. This means QoS must be implemented in the networking radio itself (see [2]).

Future studies also need to address load balancing. If there are multiple paths through the subnet, then it may be advantageous to be able to distribute the load across all the available links. However, basic research still needs to be done on this topic. Choices include equal and unequal cost load balancing, static and dynamic load balancing, and recovery from failure.

7. Conclusions

Military networking devices, such as JTRS-like radios, are often designed by a multidisciplinary team consisting of government and industry management, radio, electrical, and (hopefully) networking engineers. The management of, i.e., the coordination of, and communications between, individual members of multidisciplinary teams can be time consuming and expensive. In addition, the use of test ranges and test platforms can also be time consuming and expensive. Therefore, the ability of a single designer, or a small team of designers, to develop and debug routing protocols using simulation-assisted router protocol development environments, rather than field trials, means development can be both faster and less costly.

8. Acknowledgements

We wish to thank BAE Corporation for providing the description of the operation of the VRC-99A/B radio.

9. References

[1] Y. Rekhter, D. Kandlur: "Local/Remote Forwarding Decision in Switched Data Link Subnetworks," RFC 1937, May 1996, page 2.

[2] T.R. Henderson, P.A. Spagnolo, and J.H. Kim: "A Wireless Interface Type for OSPF," IEEE MILCOM 2003 (to appear).

[3] http://jtrs.army.mil/

[4] J.T. Moy: OSPF Anatomy of an Internet Routing Protocol, Addison-Wesley, Reading Massachusetts, 1998.

[5] A.M. Farley, P. Fragopoulou, D. Krumme, A. Proskurowski, and D. Richards: "Multi-Source Spanning Tree Problems" Journal of Interconnection Networks, Volume 1, Number 1, pp. 61-71, March 2000.

MARK STELL, Ph.D. is a network engineer at SPAWARSYSCEN San Diego. For eight years he was a member of the Navy's ADNS network design team. His current focus is developing simulation environments to support advanced development of military data communications devices.

CAM TRAN, Ph.D. is a scientist at SPAWARSYSCEN San Diego. Cam is currently developing communications device models and Operational Facilities (OPFACs) for the Navy's NETWARS effort. He is a member of the NETWARS Requirements WIPT and the NETWARS CM, Testing and V&V WIPT.

ALBERT K. LEGASPI, Ph.D. is an electrical engineer at SPAWARSYSCEN San Diego. He is currently the Navy OPNAV N61M lead on C4ISR modeling.