Poster: A Robust Header Compression Technique for Wireless Ad Hoc Networks

Ranjani Sridharan
University at Buffalo, (SUNY)
Buffalo, NY 14260-2000
rs47@cse.buffalo.edu

Ramalingam Sridhar*
University at Buffalo, (SUNY)
Buffalo, NY 14260-2000
rsridhar@cse.buffalo.edu

Sumita Mishra
CompSys Technologies Inc.,
Amherst, NY 14228
mishra@compsystech.com

ABSTRACT

Due to the limited bandwidth of the wireless links in ad hoc networks, compressing the packet headers is an attractive method for efficient usage of the frequency spectrum and for performance enhancement. The objective of this work is to demonstrate the use of a novel end-to-end header compression technique for multi-hop wireless ad hoc networks. Our preliminary analysis shows a significant performance improvement when the proposed Routing-Assisted Header Compression (RAHC) technique is implemented in conjunction with conventional on-demand routing techniques for ad hoc networks.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Wireless Communication, Network Communication, Network Topology.*

General Terms

Algorithms, Performance, Design, Reliability, Experimentation, Verification.

Keywords

Header compression, on-demand routing, error-handling, throughput enhancement, header field categorization

1. INTRODUCTION

The basic function of any header compression technique is to reduce the packet size transmitted over a link thereby increasing the link efficiency and throughput. Header compression is particularly useful in slow links where a reduction in the packet size results in a significant increase in the throughput. The idea behind any header compression technique is to avoid transmitting redundant information whenever possible. Hence, an upstream node needs to send only a short index identifying the previously sent header instead of the full header. The header compression performed in the upstream and the downstream node could go out of synchronization. Therefore, the technique should be robust to prevent the propagation of the errors to the upper layers. One of the earliest examples of header compression is that incorporated in the Thin-wire in 1984 [1]. It used a simple 20-bit field to specify which of the header's 20 bytes have changed in value. The other popular header compression techniques are: Van Jacobson's Header Compression, Space communication Protocol specification, IP Header compression, RTP Header Compression, Robust Header Compression and Unified Header Compression Algorithm [2, 3 and 4]. Due to the following limitations these techniques cannot be directly implemented into wireless multi-hop ad hoc networks:

- The existence of multiple paths between routing neighbors can result in packet re-ordering.
- In a multi-hop network, data has to be compressed and decompressed at every intermediate router, which would add to the processing overhead in the network.
- Due to the inherent nomadicity of nodes in wireless ad hoc networks, the state information is updated as nodes move out of range or fail.

As the network expands, the process overhead becomes more and more significant and also the number of compressions and decompressions increase. This results in significant delays at each router, which affects the effective data rate. The Routing Assisted Header Compression (RAHC) algorithm addresses the concerns listed above in order to optimize its performance for the targeted networks. The RAHC header compression technique relies on the information provided by the routing algorithm for wireless ad hoc networks. In order for the header compression to identify which fields are likely to change, we can assume, without loss of generalization that the header fields can be classified into one of the commonly used categories namely, constant or static fields, delta fields, inferable fields and random fields

Constant: These fields do not change among headers in a particular protocol session. These include the source and destination address.
Delta: These represent fields that change by a small amount and may not change every time. An example of the delta field is the TCP sequence number.
Inferable: These fields are determined based on other known information. For example, the packet length field can be determined from the underlying framing layer.
Random: These fields do not have a regular pattern and are best sent unchanged, *e.g.* checksums.

Once all the fields have been classified according to the above strategy, we focus on the header compression technique to avoid sending redundant information in a particular packet stream. Essentially, only those fields that are required to reconstruct the original header are sent, namely the *delta*, *inferable* and *random* fields.

* Corresponding author

MobiHoc'03, June 1-3, 2003, Annapolis, Maryland, USA.

2. COMPRESSION TECHNIQUE

Once communication is established between the sender and the receiver, the sender sends the uncompressed header with the first packet to the receiver. The receiver stores the relevant context information required to reconstruct the compressed headers that follow. The sender updates the receiver with the context information after specified intervals to inform the receiver of any changes in the static fields. This is done in order to ensure that the sender and receiver are always synchronized. Any errors in synchronization might lead to an incorrect decompression, which will pass undetected to the upper layers in the network.

In the proposed header compression algorithm, we adopt partial flooding of context information. All the nodes that lie in-between the source and the destination are updated with the context information depending on the current topology of the network and the path set up by the routing algorithm. Once a route between the source and the destination is set up, the first packet that the sender forwards will be an uncompressed packet. The receiver stores the static information from the uncompressed packet in order to be able to reconstruct the compressed packets that follow.

Since ad hoc networks are inherently nomadic, the failure of intermediate nodes might lead to de-synchronization of the source and destination nodes. In this case, the on-demand routing algorithm initiates automatic route discovery. Once the new path is identified, the context information has to be updated in all the nodes that have been newly introduced into the path. Since on-demand routing protocols maintain the route until all nodes are active, a periodic refresh of context information is not required. It is expected that the overhead cost for this update will be clearly offset by the overall bandwidth savings achieved by header compression. We are currently performing simulations to demonstrate this using the New Simulator version 2 (NS-2) [5] for preliminary simulations. We chose to use NS-2, as it is an easily available public domain tool, which fully supports wireless ad hoc networks. We have simulated a 50-nodes fully mobile network scenario using NS-2. We have observed the throughput for UDP connections between 3 independent pairs of source and sink. Figure 1 shows the throughput observations of the above network. Our preliminary studies show a significant increase in throughput on the same network with the proposed header compression technique.

3. ERROR HANDLING

One of the challenging tasks of any header compression algorithm is efficient error handling so as to reduce the impact of the errors on the upper layers of the network model. A corrupt packet will not only result in the loss of the packet but will also affect the future packets due to the maintenance of the state information. The two main areas, which may affect the future fields, are the *delta* fields and the state information. For connection-oriented protocols such as TCP/IP, errors in the reception of a packet can be resolved by retransmits. Since there is no feedback in connectionless protocols such as the UDP, the state can be maintained through periodic updates of the context information. This might decrease the throughput to some extent but it is expected that this will not be significant as compared to the case when a large number of packets are lost due to de-synchronization. We will be exploring the various causes for de-synchronization through the course of the entire development of the compression software and applying various resynchronization strategies while making sure that the degradation in throughput enhancement is minimal.

4. CONCLUSIONS

The RAHC header compression technique promises to enhance the throughput in wireless ad hoc networks. The performance of the algorithm will be analyzed based upon the results obtained via simulation. Based on these results, potential changes and enhancements will be made to the technique in order to improve the overall efficiency of header compression.

5. ACKNOWLEDGMENTS

This research was supported in part with funds from New York State office of Science, Technology and Academic Research (NYSTAR) through Micro-electronic Design Center (MDC).

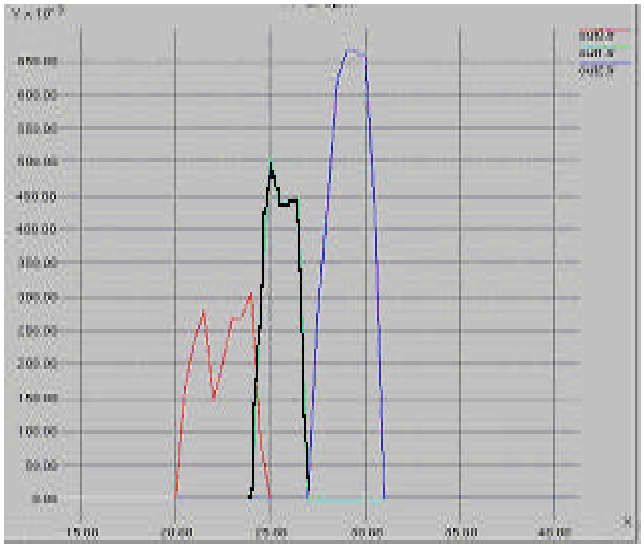

Figure 1. Throughput observation for a fully-mobile network

6. REFERENCES

[1] D. J. Farber, G. S. Delp, and T. M. Conte, “RFC-914: A Thinwire Protocol for connecting personal computers to the INTERNET”, University of Delaware, September 1984.

[2] J. Ishac, "Survey of Header Compression Techniques", Technical Report TM-2001-211154, NASA Glenn Research Center, September 2001

[3] Network Working Group, “RFC-3095: Robust Header Compression”,http://www.ietf.org/rfc/rfc3095.txt, July 2001

[4] J. Lilley, J. Yang, H. Balakrishnan, and S. Seshan, “A Unified Header Compression Framework for Low-Bandwidth Links” Proc. of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking, August 2000

[5] “The Network Simulator NS2”, http://www.isi.edu/nsnam/ns