Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Networks 50 (2006) 1349–1383

Computer Networks

www.elsevier.com/locate/comnet

MuSeQoR: Multi-path failure-tolerant security-aware QoS routing in Ad hoc wireless networks ☆

T. Bheemarjuna Reddy [a], S. Sriram [b], B.S. Manoj [c], C. Siva Ram Murthy [a,*]

[a] *Department of Computer Science and Engineering, HPCN Lab, BSB 358, Indian Institute of Technology Madras, Chennai 600 036, Tamil Nadu, India*

[b] *Department of Computer Science, University of California, Berkeley, CA 94720, USA*

[c] *CalIT2, Jacobs School of Engineering, University of California, San Diego, CA 92093, USA*

Received 2 June 2004; received in revised form 5 May 2005; accepted 23 May 2005
Available online 24 August 2005

Responsible Editor: F. Cuomo

Abstract

In this paper, we present MuSeQoR: a new multi-path routing protocol that tackles the twin issues of reliability (protection against failures of multiple paths) and security, while ensuring minimum data redundancy. Unlike in all the previous studies, reliability is addressed in the context of both erasure and corruption channels. We also quantify the security of the protocol in terms of the number of eavesdropping nodes. The reliability and security requirements of a session are specified by a user and are related to the parameters of the protocol adaptively. This relationship is of central importance and shows how the protocol attempts to simultaneously achieve reliability and security. In addition, by using optimal coding schemes and by dispersing the original data, we minimize the redundancy. Finally, extensive simulations were performed to assess the performance of the protocol under varying network conditions. The simulation studies clearly indicate the gains in using such a protocol and also highlight the enormous flexibility of the protocol.

Keywords: Multi-path routing; QoS; Security; Dispersity routing; Diversity coding; Erasure channel; Corruption channel; Ad hoc wireless networks

☆ This work was supported by the iNautix Technologies India Private Limited, Chennai, India and the Department of Science and Technology, New Delhi, India.

* Corresponding author.

E-mail addresses: arjun@cs.iitm.ernet.in (T. Bheemarjuna Reddy), sriram_s@cs.berkeley.edu (S. Sriram), bsmanoj@ucsd.edu (B.S. Manoj), murthy@iitm.ac.in (C. Siva Ram Murthy).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2005.05.035

1. Introduction

An Ad hoc wireless network is a collection of autonomous nodes that communicate with each other by establishing multi-hop paths. They are characterized by the absence of any form of fixed infrastructure. There is no centralized co-ordination among the participating nodes. Ad hoc wireless networks are primarily used for military purposes and emergency relief operations which demand provision of real-time guarantees on the delivery of data. With the increasing use of such infrastructure-less networks to handle real-time data, the question of QoS guarantees arises.

Ensuring reliable communication in Ad hoc wireless environments is a non-trivial problem. This stems from factors such as mobility of the nodes, lack of centralized control, and constraints on resources such as bandwidth and battery power. Due to the inherent broadcast nature of the medium, there is the additional concern of security of the transmitted data. To ensure reliable communication, one of the common paradigms involves setting up multiple paths between the source and the destination with reservation of resources along the paths. However, setting up of multiple paths involves a higher resource overhead. A QoS routing protocol that aims at the establishment of multiple paths must attempt to minimize the number of paths setup while ensuring that fault-tolerance is retained. Setting up of multiple paths has certain desirable characteristics in terms of the security of the data being transmitted. The security issues in multipath routing need to be explored.

We outline a multi-path QoS routing protocol that ensures reliable communication in the event of multiple path failures. The protocol provides reliability while ensuring that the number of resources reserved is the minimum required. We quantify the overhead involved for a given failure model. In addition, we quantify the security provided by the protocol. The two key issues of reliability and security are what the protocol attempts to simultaneously address. The scheme proposed is adaptive as the failure model and the number of paths to be setup are decided depending on the current state of the network and the nature of the paths available between the source and the destination. Reliability is addressed against the backdrop of both erasure and corruption channels. We initially address the issues at a conceptual level and then proceed to describe the aspects of engineering such a protocol. In this work, we have attempted to:

- Put forth a protocol that guarantees reliable communication in the face of multiple path failures and presence of adversary nodes.
- Relate the reliability provided by the protocol to the user's notion of reliability.
- Quantify the security of the protocol and relate it to the protocol parameters.
- Examine the performance of the protocol.

The rest of the paper is organized as follows: Section 2 provides a background to MuSeQoR while Section 3 discusses related work. Section 4 provides an analytical description of MuSeQoR, and Section 5 provides a description of the mechanism of the protocol. Section 6 discusses the experiment and simulation results, and Section 7 summarizes the work.

2. Background

Fault-handling in networks is based on Forward-Recovery (Hot standby) which uses redundancy to transparently handle faults without service disruption and on Detect-and-Recover (Cold standby) where the error-handling is invoked only in the event of error detection. For critical real-time applications, service disruptions cannot be tolerated and hence, the former method is preferred. Forward-recovery approaches are characterized by two parameters: *Redundancy* by which the packets can be transmitted in a manner so as to recover any lost packets, and *Dispersion* by which packets may be split and transmitted over multiple paths to the destination. These parameters are best reflected in the following two routing paradigms:

- *Redundant routing*: Establish m paths each of bandwidth B. For transmitting a volume of data D, D is duplicated on all m paths. This paradigm ensures fault-tolerance since failure of even $(m-1)$ paths still does not disrupt the communication. On the other hand, it suffers from very low throughput due to huge data overhead. In this case, the data overhead is of the order of $(m-1)$ due to the duplication of data on the m paths. In addition, this approach suffers from low call acceptance rates due to the fact that any communication requires the setting up of m paths each of which must have the required bandwidth B.
- *Dispersity routing*: Establish k paths each having bandwidth requirement B_i, $0 \leqslant i \leqslant k-1$, such that $\sum_{i=0}^{i=k-1} B_i = B$. In the degenerate case, this scheme would reduce to setting up m paths each of bandwidth $\frac{B}{m}$. A volume of data D would be split into m parts each of which would be transmitted on a separate path. This scheme would have an improved call acceptance rate since it can attempt to route along paths which do not have a bandwidth of B but do have a bandwidth of at least $\frac{B}{m}$. The lower data overhead, and consequently better resource utilization leads to higher throughput. Yet, the scheme fails to provide fault-tolerance. Failure of any one path would require retransmission of the lost packets.

The resource constraints of Ad hoc wireless networks place a premium on resource usage. Thus, a protocol that ensures reliable communication while at the same time minimizing resource usage is desirable. This is achieved by combining dispersity with some form of redundant routing. Faults in Ad hoc wireless networks occur due to link breakages and node failures: the former being a result of mobility and the latter due to the limited battery power at the nodes. In addition to such faults, there is the added possibility of malicious nodes corrupting the data packets being transmitted through them. Finally, the local broadcast nature of the channel allows nodes that are not on the path of data transmission to listen to the data. This is in sharp contrast to wired networks where the communication is over point-to-point links. Naturally, we would like to minimize the number of nodes that can listen to data not intended for them. A quantification of the security of the protocol is desirable. We explain how reliability (both in the sense of recovering lost packets and correcting corrupted packets) and security can be related to the protocol parameters.

2.1. Scenarios

We illustrate the concepts of redundancy, dispersity and the advantage of using coding schemes to achieve reliability with a few examples. In Fig. 1(a)–(e), various scenarios are depicted in the establishment of a session of bandwidth B between a source S and a destination D. Fig. 1(a) is the case of use of a single-path between S and D. Needless to say, such a setup is unreliable in an Ad hoc wireless environment. Fig. 1(b) depicts the use of a hot standby. In this case, the standby involves an additional path of bandwidth B between S and D that duplicates the data on the primary path. This arrangement addresses the reliability issue to an extent though at the cost of increased overhead. Fig. 1(c) indicates dispersity routing which has better resource utilization though not fault-tolerant. Fig. 1(d) and (e) indicate the scenarios where dispersity and coding are used to ensure reliability. In Fig. 1(d) and (e), the data to be transmitted is divided into two parts (dispersion). The actual data transmitted is some coded form of the original data and this data is sent over three paths in 1(d) and over four paths in 1(e). The scenarios in 1(d) and (e) are resilient to non-receipt of data on one of the paths and corruption of data on one of the paths, respectively.

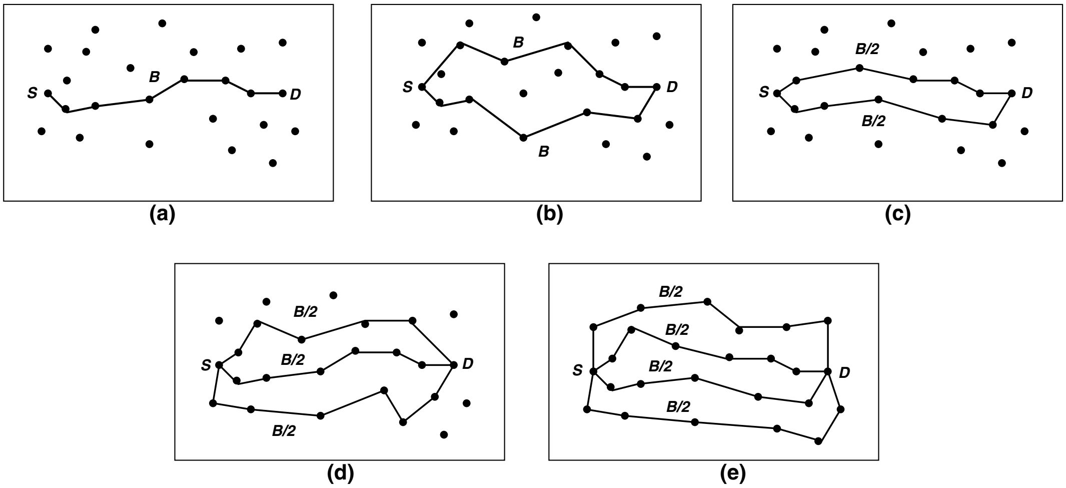


Fig. 1. Various scenarios in setting up a communication session between a source S and a destination D: (a) depicts a uni-path session, (b) depicts the use of a hot standby, (c) shows dispersity routing, and (d) and (e) depict the use of dispersity and coding in the case of erasure and corruption channels, respectively.

2.2. Objectives and motivation

Our aim is to design a QoS routing protocol that:

- Ensures reliable communication in the event of failure of multiple paths and in the presence of untrustworthy nodes.
- Enhances the security of the communication.
- Has minimum data redundancy.

3. Related work

Multi-path routing schemes are used to achieve various objectives such as balancing load, increasing reliability, improving security, and reducing the end-to-end delay. A classification of some of the existing schemes is shown in Fig. 2. There have been interesting theoretical analysis of multi-path routing [1–3]. In [4], certain criteria for the selection of multiple paths are described: namely node-disjointedness, minimum hop-length variation between primary and backup paths, and small correlation factor between the node-disjoint paths (correlation factor refers to the number of pairs of nodes on two different paths that lie within each others' transmission range). In [5], the impact of resource reservation on multi-path QoS routing is studied.

Multi-path routing aids load balancing and network utilization by distributing the load on different paths. The schemes [6,7] achieve this by using a load balancing heuristic based on the Round Trip Time (RTT) of a path. In [8], maximally-disjoint paths are setup between source and destination. In [9], on-going calls are re-routed through more optimal (often shorter) paths leading to lower end-to-end delay and better network utilization. An on-demand algorithm that uses delay and bandwidth as QoS constraints and attempts to minimize the maximum link utilization has been put forth in [10]. These protocols are on-demand schemes. Multi-path extensions to other routing protocols such as distance-vector and location-based

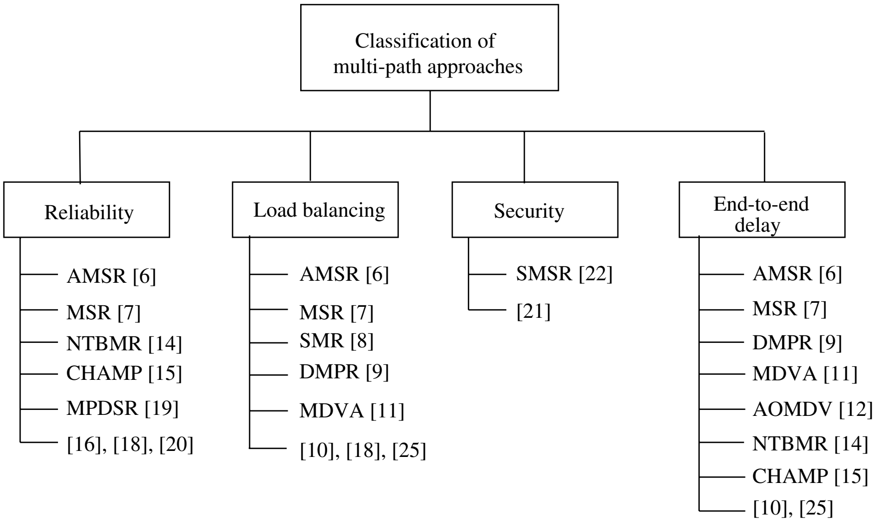


Fig. 2. Classification of multi-path routing approaches.

protocols have also been proposed [11–13]. In [14], beacons amongst two-hop neighbors are used to propagate neighbor information. Paths to a destination are chosen based on the attributes of the route such as the time elapsed since the route was setup and the route setup time. Caching and Multi-path routing Protocol (CHAMP) [15] makes use of cooperative packet caching. The nodes buffer data packets and an upstream node uses this buffer to salvage dropped packets reducing the recovery time.

Reliability is tackled by transmitting some form of redundant data over multiple paths. In [16], intermediate nodes function as regenerating nodes. Lost packets are recovered at these nodes by using FEC codes. Diversity coding [17] is one of the approaches to transmit data reliably on multiple paths. It assumes that all paths behave as erasure channels. This approach encodes blocks of m-bits to be sent on k paths as $(k + b)$ m-bit blocks, where $b < k$ is equal to the number of path failures to be protected against. These encoded blocks are then transmitted along $(k + b)$ paths. The size of each such block m is logarithmic in $(k + b)$. Tsirigos and Haas [18] studied the allocation of blocks of data to multiple paths using diversity coding such that the probability of successful transmission is maximized. This approach also works on the assumption that the channel behaves like an erasure channel. In strategic applications, not all the mobile nodes may be trustworthy. In such a scenario, one cannot assume that all the received packets are correct. Further, errors in transmission may lead to errors in the packets. In addition, in [18] the manner in which the probability of success of a path is calculated is crucial i.e., the factors that are taken into account in this calculation. Besides, the user may not always require the maximum probability of success but simply a guarantee that the established connection satisfies his reliability requirements. In addition to his reliability requirement, he may require certain other constraints such as security to be met. Leung et al. [19] links user reliability requirement to path availability. The number of paths setup for a session is determined accordingly. Call setup involves attempting to first setup a single path of required reliability. If unsuccessful, the number of paths to be setup is incremented. This may involve high call setup times. The number of paths is determined solely based on the reliability requirements. In [20] Rabin's Information Dispersal Algorithm (IDA) is used to construct a framework for a reliable multi-path scheme. It relates the probabilities of paths intersecting to the probability of failure of any of the paths used. It has not characterized the security of the scheme used.

Multi-path routing schemes that enhance the security have been proposed: a (T, N) secret-sharing scheme is proposed in [21] in which a packet is divided into N parts so that on receiving at least T packets, the data can be recovered while from any $(T-1)$ or less packets, it is computationally impossible to recover the original data; in [22], multi-path routing is used to lower the number of packets overheard by an intermediate node compared to uni-path DSR [23]. Many multi-path routing protocols establish multiple paths where only the primary path is used while the remaining paths operate in cold standby. These schemes use a detect-and-recover approach to deal with faults. Such an approach becomes unacceptable for applications in which losses cannot be tolerated. Such a multi-path extension to DSR is proposed in [24]. Das et al. [25] proposed spatial and temporal computation of a stable set of multiple paths. The multiple paths are used to disperse the data and are not used for the purpose of fault-recovery. Neither are any reservations made along the paths. Multi-path routing also needs to address scalability for deployment in large networks. In [26], a multi-path QoS algorithm is introduced that uses a concept similar to Differentiated Service [27]. A multi-path scheme that focuses on networks composed of domains has been proposed in [28].

Unlike the existing set of protocols, MuSeQoR tackles the two important issues of reliability and security, with minimum data redundancy.

4. MuSeQoR: an analytical description

We first discuss the channel and fault models that we use for the scheme. Then we provide an overview of the scheme. We provide an analysis of the manner in which multiple paths are chosen and then the security is quantified. We describe the protocol in detail in the subsequent section. Tables 1 and 2 provide a summary of the symbols that we will be using in the subsequent description and analysis.

4.1. Channel model

The channel models depend on the nature of the faults that we are attempting to tackle. We consider two channel models:

- *Erasure channels*: In erasure channels, faults are manifested in the non-arrival of packets (erasures). Packets that arrive can be assumed to be correct. This is the case when packet losses are caused by failures at links and nodes. We define an f-erasure channel as one in which at most f of the channels (node-disjoint paths) can fail simultaneously.
- *Corruption channels*: In corruption channels, in addition to the erasures, packets delivered may be corrupted by malicious nodes. An (f, g)-corruption channel is one in which at most f erasures and g corruptions can occur simultaneously.

Table 1
Summary of the symbols used to describe the protocol

Symbol	Description
S, D	Source, destination of a session
B	Total bandwidth of the session
k	Dispersion factor
f	Maximum number of simultaneous erasures
g	Maximum number of corruptions
b	Total number of backup paths
n	Total number of paths setup between S and D
ϵ	The path-failure metric
ER	Eavesdropping ratio

Table 2
Summary of the symbols used in the analysis

Symbol	Description
$P_i(\delta t)$	Probability of failure of i paths in the time interval $(t, t+\delta t)$
$P_{(0,\ldots,i-1)}(\delta t)$	Probability of failure of paths $0,\ldots,i-1$ in $(t, t+\delta t)$
$A_{(i,j)}(\delta t)$	Link availability of the link (i,j) in $(t, t+\delta t)$
$B_i(\delta t)$	Availability of node i in $(t, t+\delta t)$
$M=(M_1,\ldots,M_k)$	k packets of data to be sent
$C=(C_1,\ldots,C_n)$	n packets of coded data actually sent
$G=[g_{ij}]$	Mapping matrix between vectors C and M
C'	The set (vector) of packets received at D
m	Bits in each packet in M and C Also the field in which coding is done
$N(u)$	Neighbor set of node u
$T(P_1,\ldots,P_n)$	Set of eavesdroppers which can listen to all the packets of a batch being transmitted on the paths $P_1,\ldots,P_n$
$T_m(P_1,\ldots,P_n)$	Set of eavesdroppers which can listen to at least m of the packets of a batch being transmitted on the paths $P_1,\ldots,P_n$
$T_{uni\text{-}path}$	Set of eavesdroppers on the shortest uni-path route between S and D
$X(u)$	Event that node u can listen to at least k packets of the (S, D)-session
$Y(u)$	Event that a node u is a neighbor of S

We term these events as *adversary events* in the channel. In erasure channels, the adversary events are the path failures. In corruption channels, the adversary events are path failures and corruptions.

4.2. Fault model

Given a set of n node-disjoint paths, let $P_i(\delta t)$ denote the probability of failure of i $(0 \leqslant i \leqslant n)$ of these n paths in a time-interval $(t, t+\delta t)$. We say that the set of n node-disjoint paths follows an f-path failure model $(f < n)$, if for a given $0 \leqslant \epsilon \leqslant 1$, $P_f(\delta t) > \epsilon$ and $P_{f+1}(\delta t) < \epsilon$ where ϵ is a path-failure metric. ϵ is an upper limit on the failure of a given session that can be tolerated by an application. By failure, we refer to non-delivery or corruption of data packets on the data paths. A low value of ϵ implies a lower likelihood of simultaneous failure of paths and hence, a higher reliability requirement. We look at ϵ as a specification of the application: a time-critical or data-critical application would specify a lower value of ϵ as opposed to a delay-tolerant or best-effort application.

We assume that node-disjointedness is sufficient to assume that the events of failure of the individual paths are independent. Thus, the probability of failure of the paths $p_0,\ldots,p_{i-1}$ in the interval $(t, t+\delta t)$ is given by

$$P_{(0,\ldots,i-1)}(\delta t) = \prod_{l=0}^{l=i-1} P_{p_l}(\delta t),$$

where $P_{p_l}(\delta t)$ is the probability of failure of the path p_l in that interval.

Path failures occur due to node failures that are a result of lack of battery power and due to link failures caused by mobility.

We can write the path failure probability for a path p_l as

$$P_{p_l}(\delta t) = 1 - \prod_{(i,j)\in p_l} A_{(i,j)}(\delta t) \prod_{i\in p_l} B_i(\delta t),$$

where $A_{(i,j)}(\delta t)$ is the link availability of the link connecting nodes i and j on path p_l as discussed in [29] and $B_i(\delta t)$ is the node availability of the node i on path p_l. Link availability $A_{(i,j)}(\delta t)$ is the probability that there exists an active link between nodes i and j at time $t + \delta t$ given that there is an active link between them at time t. Node availability $B_i(\delta t)$ is defined similarly and accounts for the failure of nodes due to power dissipation (*Note*: Independence of link failures and node failures is used only to compute the path failure probability. We could use a more sophisticated model to compute path failure probabilities from the link and node failure probabilities. Once the path failure probabilities are known, we can consider the possibility of failure of multiple paths). Thus, we can rewrite the probability of failure of the paths $p_0, \ldots, p_{i-1}$ in the interval $(t, t + \delta t)$ as

$$P_{(0,\ldots,i-1)}(\delta t) = \prod_{l=0}^{l=i-1} \left(1 - \prod_{(i,j)\in p_l} A_{(i,j)}(\delta t) \prod_{i\in p_l} B_i(\delta t) \right). \tag{1}$$

4.3. Overview

We need to establish a session of bandwidth B between source S and destination D so that the paths can sustain communication in the face of adversary events corresponding to the channel model. We do this by setting up n node-disjoint paths each of bandwidth $\frac{B}{k}$, $n \geqslant k$. The use of node-disjoint paths offers two distinct benefits: it makes the assumption of independent failures of the paths more credible, and secondly, it improves the security of the session by ensuring that none of the intermediate nodes have more than a single path passing through them (although they can possibly eavesdrop). Thus, the node-disjointedness has inherent benefits and in this work we show this approach to be viable. If the node were to be a high-bandwidth node, it would still be possible to use a multi-path scheme by sending different number of packets along the paths as opposed to the present scheme of sending the same number of packets on each path. Further, in a typical Ad hoc setting, we assume that all the nodes have the same capabilities in terms of data generation and packet forwarding so that the relationship between any two nodes is that between peers. Hence, we do not consider this design issue. The packets are sent by splitting the message into k m-bit blocks. We transform the k blocks into n blocks using an (n, k)-code and then these n blocks are transmitted over the n paths setup. The (n, k)-code ensures that the destination can recover the original k blocks in the case of adversary events. A session that uses an (n, k)-code is referred to as an (n, k)-session.

We need to determine the minimum number of paths to be setup i.e., the value of n so that the connection formed by the set of paths established is reliable for the given channel and fault model and a given value of k. We do this analysis for the cases of the erasure and corruption channels. We describe existing coding techniques that we use to ensure reliable routing.

4.4. Erasure channel

Consider an f-erasure channel. From the theory of erasure channel codes, the minimum value of n is given by $n = k + f$ provided $2^m > n$. Let $M = (M_1, \ldots, M_k)$ be the vector representing the actual data to be sent. Each of $M_1, \ldots, M_k$ is an m-bit block. The data transmitted is represented by the vector $C = (C_1, \ldots, C_n)$ where $C_1, \ldots, C_n$ is an m-bit coded block. All operations are performed in the field $\mathbb{GF}(2^m)$ since each element of the vector is an m-bit number (a finite field is a finite set of elements with two operations, named addition and multiplication, defined on them. These operations have inverse operations and additional properties such as closure, associativity, and presence of identity. In addition, multiplication must distribute over addition. The order of the field is the number of elements in its set. Thus, $\mathbb{GF}(2^m)$ represents a field of order 2^m. Every finite field has an element α that can generate all the elements of the field except the additive identity. For more information, refer [30,31]). The code vector is related to the data vector as $C = MG$ where G is a $(k \times n)$ matrix whose elements belong to $\mathbb{GF}(2^m)$.

At the destination, some subset of the n coded blocks arrive. For an f-erasure channel, at least $n-f$ blocks are expected to be received. If C' is the vector of some set of k blocks received and G' is the sub-matrix formed by the subset of columns of G corresponding to the blocks of C', then, we have $C' = MG'$ and $M = C'G'^{-1}$ where G'^{-1} is the inverse of G'.

Thus, to recover the original data M in an f-erasure channel, the sub-matrix formed out of every subset of $n-f$ columns of G must be invertible. Let α be the generator of $\mathbb{GF}(2^m)$. The following construction of G:

$$G = [g_{ij}] = [\alpha^{(i-1)j}] \quad \text{where } 1 \leqslant i \leqslant k,\ 1 \leqslant j \leqslant n \tag{2}$$

satisfies the property. Any subset of k columns of G is a Vandermonde matrix whose determinant is non-zero provided all the α^j's are different. This is the case when $2^m > n$.

For example, for $m = 3$, $\mathbb{GF}(2^3)$ can be thought of as the field of polynomials with degree at most two with coefficients drawn from $\mathbb{GF}(2)$ (0 or 1). Such a field has elements of the form

$$0,\ 1,\ x,\ x+1,\ x^2,\ x^2+1,\ x^2+x,\ x^2+x+1$$

with multiplication done, say, modulo x^3+x+1. For $\mathbb{GF}(2^3)$, x is a generator (i.e., we may set $\alpha = x$). Thus, the non-zero elements may be written in terms of powers of the generator as

$$1 = \alpha^0,\ x = \alpha^1,\ x+1 = \alpha^3,\ x^2 = \alpha^2,\ x^2+1 = \alpha^6,\ x^2+x = \alpha^4,\ x^2+x+1 = \alpha^5.$$

We can construct a (5, 3) code, since $2^3 > 5$, given by the matrix G

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ x & x^2 & x+1 & x^2+x & x^2+x+1 \\ x^2 & x^2+x & x^2+1 & x & x+1 \end{pmatrix}.$$

Thus, in an f-erasure channel, it is sufficient to setup $k+f$ paths between the source and the destination to ensure reliable communication. The redundancy ratio RO defined as the ratio of the amount of data sent in excess to the original data, in this case, is

$$\mathrm{RO} = \frac{f}{k}. \tag{3}$$

4.5. Corruption channel

In a corruption channel, in addition to packet losses, we need to deal with the delivery of incorrect packets. This can be done by the use of codes such as the Reed–Solomon codes. The Reed–Solomon codes are based on the Fourier transform. An (n, k) Reed–Solomon code can detect and correct f erasures and g failures provided $n - k \geqslant 2g + f$ [32]. Thus, for a (f, g)-corruption channel, we need to setup a minimum of $k + f + 2g$ paths to ensure reliable communication. The redundancy ratio in this case is

$$\mathrm{RO} = \frac{2g+f}{k}. \tag{4}$$

Some of the issues with respect to providing reliability are:

- It is possible to use Reed–Solomon codes for the case of erasure-channels as well. However, the Vandermonde matrix-based erasure-channel codes are easier to decode.
- In the cases of both the erasure and the corruption channels, the codes used may be systematic (i.e., k out of the $n = k + b$ paths carry the unencoded data) without any change in the scheme. However, a non-systematic code that ensures that all paths carry encoded packets is attractive from the point of view of security.

- The parameter f is determined by the user requirements whereas the parameter g is fixed depending on the state of the network.

4.6. Security

The notion of security that we consider has to do with the fact that the data being transmitted can be overheard by nodes other than those on the $n = (k + b)$ paths. This is due to the local broadcast nature of the medium. In this notion of security, we assume the adversaries are passive and non-colluding. Improving the security is done by reducing the number of nodes that can listen to the data. Our protocol enhances security by dispersity and coding. Dispersity results in the data being scattered over a wider area of the network, thereby reducing the probability that a node can overhear all the packets being transmitted. On the other hand, dispersity also increases the number of nodes that can access some portion of the data. The mapping that is used to code the packets is shared between the source and the destination by means of a Public-key cryptosystem. Since adversaries do not have the mapping information, even if they have access (by overhearing) to the packets transmitted on a certain fraction of the paths, it would be difficult to obtain the entire data.

We introduce a metric, the eavesdropping ratio (ER): a measure of the leakage of data due to the routing protocol. Consider a session between two nodes occurring over n paths $P_1, \ldots, P_n$. Define a batch of n packets as the set of n packets formed by encoding the k packets to be transmitted. For successful decoding of the packets at the destination, it is necessary that at least k packets in each batch must reach the destination. For any eavesdropping node to have a successful chance of decoding the coded packets, it must be able to eavesdrop at least k of the packets in a batch.

Denote by

$N(u)$: the neighbor set of node u i.e., the set of all nodes lying within the transmission range of u.
$T(P_1, \ldots, P_n)$: the set of nodes which can listen to all the packets of a batch being transmitted on the paths $P_1, \ldots, P_n$.
$T_m(P_1, \ldots, P_n)$: the set of nodes which can listen to at least $m < n$ of the packets of a batch being transmitted on the paths $P_1, \ldots, P_n$.
$N(P) = \cup_{u \in P} N(u)$ where P is a path.
For the set of paths $(P_1, \ldots, P_n)$ belonging to an (n, k)-session:

$$T(P) = N(P) \cup P,$$
$$T(P_1, \ldots, P_n) = \cap_{i \in 1,\ldots,n} T(P_i),$$
$$T_m(P_1, \ldots, P_n) = \cup_{1 \leqslant i_1 < i_2 < \cdots < i_m \leqslant n} [\cap T(P_{i_1}) \ldots T(P_{i_m})].$$

It is clear that: $|T(P_1, \ldots, P_n)| \geqslant |N(S)|$, where S is the source of the paths.

Define the *eavesdropping ratio* ER as

$$\mathrm{ER} = \frac{|T_k(P_1, \ldots, P_n)|}{|T_{uni\text{-}path}|}, \tag{5}$$

where $T_{uni\text{-}path}$ is the set of nodes that can listen to all packets on the shortest uni-path route between the source and the destination.

For example, consider the network topology shown in Fig. 3(a). A (3, 2)-session is setup (see Fig. 3(b)) by reserving bandwidth $B/2$ along three node-disjoint paths from source S to destination D.

Here

$P1 = S \rightarrow V \rightarrow C \rightarrow R \rightarrow T \rightarrow D$,
$P2 = S \rightarrow I \rightarrow W \rightarrow F \rightarrow G \rightarrow D$,

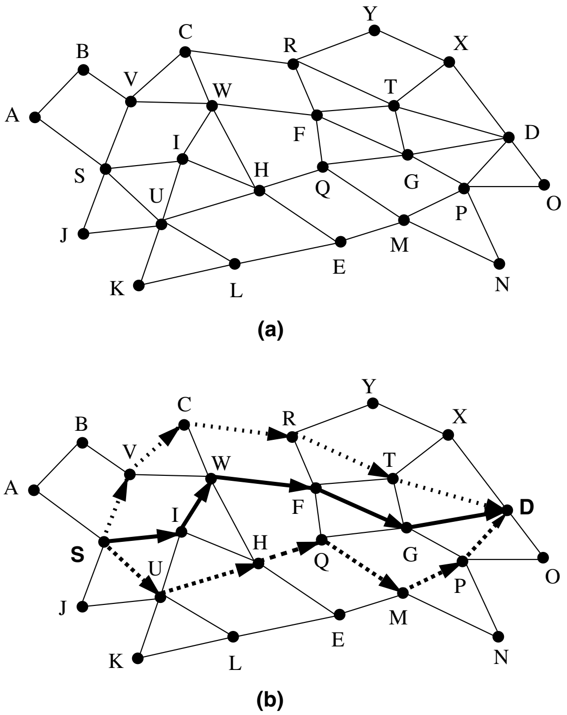


Fig. 3. A (3,2)-session between a source S and a destination D.

$P3 = S \rightarrow U \rightarrow H \rightarrow Q \rightarrow M \rightarrow P \rightarrow D$,
$T(P1) = \{S, A, J, V, I, U, B, C, W, R, F, Y, T, G, X, D\}$,
$T(P2) = \{S, A, J, V, I, U, W, H, C, F, R, T, Q, G, P, D\}$,
$T(P3) = \{S, A, J, V, I, U, H, K, L, W, E, Q, F, M, G, N, P, O, D\}$,
$T(P1, P2, P3) = \{S, A, J, V, I, U, W, F, G, D\}$,
$T_2(P1, P2, P3) = \{S, A, J, V, I, U, W, F, T, G, P, C, R, H, Q, D\}$. Let $P1$ be the shortest uni-path route between S and D. The ER in that case is $\frac{16}{16} = 1$.

The eavesdropping ratio ER is a measure of the security of the protocol and can be related to the parameters of the protocol.

4.7. Relation between ER and the protocol parameters

Consider an Ad hoc wireless network of N nodes in an area A, with the transmission range of each node being R. Consider an (n, k)-session between a source S and a destination D. Let $X(u)$ be the event that a node u can listen to at least k packets and $Y(u)$ the event that a node u is a neighbor of the source i.e., $u \in N(S)$.

$P(X(u)|Y(u)) = 1$, since every node in the transmission range of S hears all the packets being transmitted by S. For a node to listen to at least k packets, at least k of the n paths must pass through its transmission range. Since the paths are node-disjoint, this is possible only if $|N(u)| \geqslant k$.

$$P(X(u)|\overline{Y(u)}) = P(X(u)||N(u)| \geqslant k, \overline{Y(u)})P(|N(u)| \geqslant k|\overline{Y(u)}),$$
$$P(X(u)|\overline{Y(u)}) \leqslant P(|N(u)| \geqslant k|\overline{Y(u)}).$$

Using Bayes' theorem,

$$\begin{aligned} P(X(u)|\overline{Y(u)}) &\leqslant \frac{P(\overline{Y(u)}||N(u)| \geqslant k)P(|N(u)| \geqslant k)}{P(\overline{Y(u)})}, \\ P(X(u)|\overline{Y(u)})P(\overline{Y(u)}) &\leqslant P(\overline{Y(u)}||N(u)| \geqslant k)P(|N(u)| \geqslant k), \\ P(X(u)|\overline{Y(u)})P(\overline{Y(u)}) &\leqslant P(|N(u)| \geqslant k). \end{aligned} \tag{6}$$

Let p denote the probability that a node $y \in N(u)$. Then the value of $p = \frac{\pi R^2}{A}$. Also, $P(Y(u)) = p$.

$$P(|N(u)| \geqslant k) = 1 - P(|N(u)| < k),$$
$$P(|N(u)| \geqslant k) = 1 - \sum_{i=0}^{i=k-1} P(|N(u)| = i).$$

The probability that $|N(u)| = i$ is given by the probability that i of the $(N - 1)$ nodes (all nodes other than u) are in the transmission range of u and all of the remaining $(N - 1 - i)$ are outside the transmission range.

$$P(|N(u)| \geqslant k) = 1 - \sum_{i=0}^{i=k-1} \binom{N-1}{i} p^i (1-p)^{(N-1-i)}.$$

Since $P(X(u) \cap \overline{Y(u)}) = P(X(u)|\overline{Y(u)})P(\overline{Y(u)})$, from (6):

$$P(X(u) \cap \overline{Y(u)}) \leqslant 1 - \sum_{i=0}^{i=k-1} \binom{N-1}{i} p^i (1-p)^{(N-1-i)}.$$

The expected value of the event $X(u) \cap \overline{Y(u)}$ is given by

$$E(X(u) \cap \overline{Y(u)}) = (N - |N(S)| - 1)P(X(u) \cap \overline{Y(u)}).$$

Thus, the average number of nodes, E, that can listen to at least k packets

$$E = |N(S)| + E(X(u) \cap \overline{Y(u)}),$$
$$E \leqslant |N(S)| + (N - |N(S)| - 1)\left(1 - \sum_{i=0}^{i=k-1} \binom{N-1}{i} p^i (1-p)^{(N-1-i)}\right).$$

We get

$$\begin{aligned} \mathrm{ER} &= \frac{E}{|T_{uni\text{-}path}|}, \\ \mathrm{ER} &\leqslant \frac{|N(S)| + (N - |N(S)| - 1)\left(1 - \sum_{i=0}^{i=k-1} \binom{N-1}{i} p^i (1-p)^{(N-1-i)}\right)}{|T_{uni\text{-}path}|}. \end{aligned} \tag{7}$$

ER can be regarded as a relative measure of the leakage of data in a wireless network for a given multipath routing scheme to that in shortest-path uni-path routing between the source and the destination. A

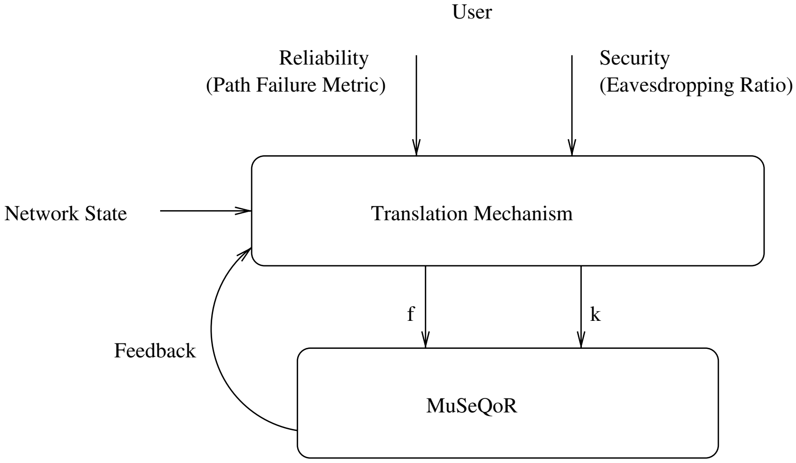


Fig. 4. Translation mechanism.

lower value of ER implies a lower leakage. The inequality gives us a way to estimate a bound on k so that a given value of ER is met. This along with the number of neighbors of the source and the destination can be used to determine the number of paths that need to be setup (*Note*: The computation of ER does not require global topology information. The value of $|N(S)|$ is communicated to the destination at the time of route setup).

The proposed protocol is characterized by the three parameters (k, f, g) in the general case. The user requirements are specified by the parameters ϵ and ER. These requirements are then translated into values for k and f (Fig. 4). g is simply the number of malicious nodes in the network and can be specified as such. Thus, the protocol tackles the twin issues of reliability and security.

5. Description of the protocol

The protocol that we propose is an on-demand protocol that is a modification of the Dynamic Source Routing (DSR) [23] protocol. Thus, no global topology information is maintained at the nodes. A session is divided into time frames Δt during which the network state is assumed to be fairly constant. The parameters computed and the paths setup during a time frame remain unaltered unless disruption of the existing paths occurs. To enhance security, each time frame is further divided into code sessions of length Δt_c. A code session is the period for which a single mapping (in this case the matrix G referred to in Eq. (2)) is used by the source for encoding the packets. This mapping may be changed at the end of a code session by means of a Public-key cryptosystem.

We must emphasize that our protocol and encryption are orthogonal, and hence a spectrum of designs are possible based on the use of encryption. At one extreme, no data or control packets are encrypted. This scheme would be a lightweight protocol. In such a scheme, a node that would like to eavesdrop must first eavesdrop and obtain the mapping, and then eavesdrop to obtain at least k packets which it decodes. Thus, the eavesdropping ratio plays a crucial role in the security of the process. At the other end of the spectrum, all the packets exchanged would be encrypted: this is however a rather heavyweight solution, and does not lend itself well to a resource-constrained ad hoc environment. Yet, in a scenario where the data being carried is critical in nature, such a scheme may be adopted. In the middle, we can use a scheme in which the mapping alone is exchanged over an encrypted channel while the actual data is sent in the clear. Encryption using a Public-key cryptosystem is used only to exchange mappings between the source and the destination. The data, itself, is not encrypted due to the heavy-weight nature of the encryption–decryption operations. On the other hand, the fact that the data is encoded (using erasure or error-correcting codes) before transmission rather than being transmitted in the clear prevents malicious nodes from obtaining any useful information by eavesdropping. Thus, the process of encoding the data has the fringe-benefit of security. This does not prevent the intermediate nodes from altering the data that they receive so that the destination receives junk. This is the case of a corruption channel and correcting these altered packets would require the use of error-correcting codes. The intermediate nodes can still attempt to decode packets by eavesdropping on a number of sessions and in this context, the eavesdropping ratio must be kept low.

5.1. Translation mechanism

The level of security and reliability provided by the protocol depends on the protocol parameters $k, f,$ and g where k is the dispersity factor, f is the number of erasures to be tolerated, and g is the number of corruptions to be protected against. The manner in which these parameters are frozen needs to be determined. We can view the operation of the protocol in the following ways:

- The parameters $k, f,$ and g are specified preferably by the user. This approach allows the user to control the level of service provided to him by the protocol. Yet such an approach assumes that the user knows the values of $k, f,$ and g that he needs. This need not be the case. What matters to the user is the level of reliability and security that the session provides him. A static specification of protocol parameters may not prove to be very beneficial to the user. A value of $f = 1$ may be inadequate in a highly mobile network while a high value of f may lead to unnecessary reservation of resources for a network which is fairly static.
- Thus, it is more meaningful for the user to specify his reliability and security requirements in terms of a more abstract metric (see Fig. 4). This metric then needs to be converted into the protocol parameters so that MuSeQoR can setup the required number of paths. The user reliability metric is related to the fraction of packets delivered successfully, while the security metric is related to the eavesdropping ratio: the ratio of the number of (non-source and non-destination) nodes that are able to listen to a specified fraction (given by $\frac{k}{k+b}$) of the packets in each batch in the multi-path routing strategy to the same number in the shortest path uni-path routing. The number of corruptions to be protected against is specified in a straightforward manner as the number of malicious nodes in the network. With the user specifying his requirements in this way, the translation of the user requirements into protocol parameters can be done in a dynamic manner that depends on the state of the network. This ensure that the protocol is adaptive and that the resources setup are appropriate to the state of the network environment. This translation of QoS requirements is done primarily during route setup, and, when required, during route maintenance.

5.2. Route setup and parameter selection

In this phase, paths between the source and the destination are probed. Depending on the nature of the paths found the fault-model is chosen. This manner of choosing the fault-model ensures that the protocol adapts to the state of the network. Accordingly, the dispersion factor (parameter k) and the corresponding number of backup routes (parameter b) are estimated. The protocol then attempts to find these routes. Algorithms 1–3 describe the algorithms at the source, the destination, and the intermediate nodes respectively.

Algorithm 1. Source node

```
packet PACKET;          // PACKET can be a DATA, RESERVE, or RERROR packet.
/* REQ_ATTEMPTS: It keeps track of number of route requests initiated so far by the Source node.
MAX_ATTEMPTS: Maximum number of times the Source node tries to establish the session (by
sending route requests) before rejecting the call.
session.S-CACHE: A data structure (route cache of the current session) that maintains a set of
node-disjoint paths available to the Destination node. */
PACKET ← packet received.
if PACKET is a DATA packet then
  Buffer PACKET.
  if session.S-CACHE is empty then
    minimum ← 1          //minimum number of paths to be setup
    maximum ← 1          //maximum number of paths to be setup
  end if
  SendData:
  if |session.S-CACHE| < minimum then
    if no outstanding REQUEST and REQ_ATTEMPTS < MAX_ATTEMPTS then
      Send REQUEST to Dest. REQ_ATTEMPTS++. Set REP_TIMER. /* Starts a timer and waits
      for receiving enough number of RESERVE packets to initiate the session. */
    else if REQ_ATTEMPTS ⩾ MAX_ATTEMPTS then
      Reject the call. Clear buffer.
    end if
  else
    if level is STRICT and |session.S-CACHE| < maximum then
      if no outstanding REQUEST and REQ_ATTEMPTS < MAX_ATTEMPTS then
        Send REQUEST to Dest. Set REP_TIMER.
        REQ_ATTEMPTS++
      else if REQ_ATTEMPTS ⩾ MAX_ATTEMPTS then
        Reject the call. Clear buffer.
      end if
    else
      while at least $k$ packets in the buffer do
        REQ_ATTEMPTS ← 0
        Dequeue $k$ packets, encode into $n$ packets and transmit.
      end while
    end if
  end if
else
  if PACKET is a RESERVE packet then
    minimum ← PACKET.$k$
    maximum ← PACKET.$n$
    Add PACKET.path to session.S-CACHE.
    goto SendData.
  else if PACKET is a RERROR packet then
    if no outstanding REQUEST then
      Send REQUEST to Dest. Set REP_TIMER.
      REQ_ATTEMPTS++
    end if
  end if
end if
```

Algorithm 2. Destination node

```
packet PACKET;          // PACKET can be a DATA or REQUEST packet.
/* session.D-CACHE: A route cache of the current session that maintains a set of node-disjoint
paths available to the Source node.
session.TMP-CACHE: A temporary route cache of the current session that maintains a set of
paths (not yet added into session.D-CACHE) available to the Source node. */
PACKET ← packet received.
if PACKET is a DATA packet then
  Mark the path on which the data packet arrived as fresh.
  Buffer PACKET.
  if packets received in a set $\geqslant k$ then
    Decode packets.          /* Destination has received at least $k$ coded packets in the
    current batch, hence it successfully gets the original packets by decoding the coded packets. */
    Clear buffer.
  end if
else
  if PACKET is a REQUEST packet then
    Remove stale paths from session.D-CACHE and session.TMP-CACHE.
    if new PACKET.session then
      Create session ← PACKET.session
    else
      session ← PACKET.session
    end if
    Add PACKET.path to session.TMP-CACHE.
    if parameters for session are determined and Time < $\Delta t$ then
      /* Time is the duration of the session. A session is divided into time frames $\Delta t$ during which
      the parameters computed and the paths setup remain unaltered. */
      if |session.D-CACHE| < session.$n$ then
        Check if paths in session.TMP-CACHE and in session.D-CACHE yield a feasible solution
        for the parameters $k, f, b$ and bandwidth. /* This means there exists a set of $(k + b)$ node-
        disjoint paths that satisfy the bandwidth requirements. */
        If it does, dispatch RESERVE packets along the new paths.
        Add the new paths to session.D-CACHE.
      end if
    else
      if paths in session.TMP-CACHE and session. D-CACHE allow a feasible solution to the parameters
then
        Add the paths to session.D-CACHE. /* Need to check if the paths in the session.TMP-CACHE
and session. D-CACHE are disjoint and enough in number. */
      end if
    end if
    end if
end if
```

Algorithm 3. Intermediate node

```
packet PACKET;                    // PACKET can be a DATA or REQUEST packet.
PACKET ← packet received.
if PACKET is a DATA packet then
    if this node is the next-hop then
        Forward PACKET to downstream node towards the destination.
    end if
else
    if PACKET is a REQUEST packet then
        if the number of packets seen with the same source and sequence number < MAX_FORWARD
        then
            Use information in the PACKET to compute link availability of the incoming link.
            Modify the path availability.
            Increment count of packets seen with the same source and sequence number.
            Forward PACKET towards the destination.
        end if
    end if
end if
```

Route discovery: Route request (REQUEST) packets are sent by the source to its neighbors. Each REQUEST packet carries a sequence number, the source and destination IDs, the connection ID, the path traversed so far, the reliability of the path traversed so far, the path bandwidth, the bandwidth required for this session, and the path failure metric ϵ and the eavesdropping ratio (ER) specified for this session. Each intermediate node checks the route of the packet to ensure that there are no loops. There are two possible strategies available for forwarding of packets by the intermediate nodes: the intermediate node may forward all route requests arriving at it (flooding) or it may forward only the first request of a particular sequence number discarding all future route requests carrying the same sequence number. The former strategy is more beneficial in terms of discovering a larger number of routes to the destination while the latter method reduces control overhead. We adopt an intermediate approach wherein the intermediate node forwards a maximum of MAX_FORWARD (a system parameter) route request packets. The higher the value of the system parameter, the greater the amount of flooding in the network while the number of routes discovered increases.

Before forwarding the REQUEST packet, the node multiplies the link availability of the link on which it received the packet to the reliability being carried in the packet. It also modifies the bandwidth available on the path and forwards the packet to its neighbors.

Determination of f and b: The destination receives r REQUEST packets (r must be greater than the number of paths that need to be setup else no reply is sent back). It then sorts the paths according to their probabilities of failure p_f. Let $\{P_1,\ldots,P_r\}$ be the paths where $p_f(P_1) > \cdots > p_f(P_r)$. Find $i+1$ so that $p_f(P_1) \times p_f(P_2) \times \cdots \times p_f(P_{i+1}) < \epsilon$. Then an i-path failure model is used. This calculation of i is an approximation in that the paths $P_1,\ldots,P_r$ are not node-disjoint, except when MAX_FORWARD is 1. The number of additional paths b to be setup is determined either as $b = f$ for the case of erasure channels and $b = f + 2g$ for the case of corruption channels.

Determination of the link availability: For a particular mobility model, the expression given in [29] may be used. We, however, use a more approximate method in which we also ignore the node availability B_i (assuming that the nodes do not lose power too quickly). The REQUEST packet carries the position and velocity information (as a vector so that both magnitude and direction are known) of the last node i that forwarded it. On arriving at the next node j, this information is used by the node to compute the relative distance and velocity of i. The node j then calculates the time t for which the upstream node i remains within its transmission range, assuming that they maintain the same velocities. Then the link availability $A_{(i,\ j)}$ is given by

$$A_{(i,j)} = 1 - \mathrm{e}^{\alpha t},$$

where α is a proportionality factor, and e is the natural logarithmic base.

Determination of k: k must be chosen subject to the constraint that: $k + b \leqslant \min(|N(S)|, |N(D)|)$, where $N(S)$ and $N(D)$ are the set of nodes lying within the transmission range of source S and destination D, respectively. In addition, the eavesdropping ratio ER is used to determine the value of k.

Route calculation and Resource reservation: Once k is determined, the destination finds out if there exists $(k + b)$ node-disjoint paths with bandwidth $\frac{B}{k}$. Each REQUEST packet arriving at the destination carries the path traversed. The destination waits for a certain period of time during which it receives many such packets. The destination then uses these paths and does a search to obtain a set of node-disjoint paths. It ensures that the paths satisfy the bandwidth-requirement of the session.

If it finds a set of paths, it sends a route reserve (RESERVE) packet on each of the paths. The remaining paths are stored in its route cache. The RESERVE packets carry the route, values k, f, and b and the mapping to be used for encoding for the remaining code session. On receiving the RESERVE packets on the $(k + b)$ paths, the source begins data transmission. Some of the paths on which the RESERVE packets are being sent may not have the bandwidth as other calls may have reserved it in the meantime. The node which does not possess the required resources simply drops the RESERVE packet. Resources that have been reserved thus far are released when a timeout occurs due to non-arrival of packets of the connection on that path.

The protocol has two levels of communication: a STRICT level in which the call is accepted only if n RESERVE packets are received at the source, and a LAX level in which the source can begin data transmission even if it receives RESERVE packets on m, $k \leqslant m \leqslant n$ paths. The scenario is treated as failure of the $(n - m)$ paths and route repair is initiated. In the meantime, the source can continue to send packets on the m paths. In the LAX level of operation, the call is rejected if the source receives less than k RESERVE packets. The resources reserved on the paths traversed by the RESERVE packets are released on a timeout due to non-arrival of additional packets.

If the destination is unable to select a set of node-disjoint paths that satisfy the bandwidth constraint, it does not send any RESERVE packets. The source on non-receipt of RESERVE packets re-initiates the route discovery. This process is repeated MAX_ATTEMPTS times. Failure on all occasions results in the call being rejected at the source. When the destination is unable to find the set of node-disjoint paths when the REQUEST packets are sent for the MAX_ATTEMPTS time (this is indicated in the REQUEST) packet, all the paths to the source are removed from the route cache of the destination.

5.3. Route maintenance

Route maintenance occurs in two scenarios.

Expiry of the time frame: At a time $\Delta t - 2\mathrm{RTT}$, where RTT is the round trip time of the transmission, the data packets on the various paths carry the path failure probability. At the destination, this is used to verify that the paths still conform to the f-path fault model. If they do not, the destination sends back a RREPAIR packet to the source. The source then initiates a route discovery to search for a set of paths disjoint with the current set of paths. The destination uses the newly discovered paths along with the currently used $(k + b)$ paths to recompute the parameter f. If f has increased, the destination sees if the newly discovered paths can be added to the already existing paths to meet the new reliability constraint. If this is possible, RESERVE packets are sent along the newly discovered set of paths and the data transmission continues. On the other hand, failure to establish the required number of paths will be treated in the same way as in the route setup process depending on the level of the communication (i.e., whether it is STRICT or LAX). If the value of f decreases, on the other hand, the paths are maintained as such.

Route break: If the route break occurs when the parameters are being re-estimated (during the time $\Delta t - 2\text{RTT}$ to Δt), no action is taken as anyway the routes may need to be reconfigured. Otherwise, failure of a link (u, v) where u is the upstream node results in node u sending a RERROR packet to the source. On receiving this packet the source initiates a route discovery. The resources reserved on the downstream nodes are released on a timeout due to non-arrival of any more packets.

In the protocol, both the source and the destination need to know the set of paths used for communication. The encoding and decoding of the transmitted data requires this. Every node has a route cache of paths to a particular destination node for which it is the source (S-CACHE) and another route cache of paths from a source node for which it is the destination (D-CACHE). For the communication to be successful, the paths in the route caches of the source (S-CACHE) and the destination (D-CACHE) must be consistent. After the first route request by the source, the destination selects the set of node-disjoint paths if available and inserts these into its D-CACHE. When the RESERVE packets arrive on the paths at the source, the source adds these paths to its S-CACHE. A path break causes a RERROR packet to be sent to the source node which removes the path (and all paths of any other connection that may be passing through the failed link) from its S-CACHE. It then initiates another REQUEST to the destination. At the destination, paths which have not been delivering packets (either data or control) for a time exceeding FRESH_WINDOW are marked stale. When a REQUEST arrives at the destination, if this has arrived on a path marked stale, the path is re-included into the D-CACHE. Otherwise, a new set of paths to the source that includes the existing set of non-stale paths is computed and if this set is found to satisfy the bandwidth and the reliability requirements, the stale paths are expunged and RESERVE packets are sent along the newly-discovered paths.

6. Experiment and simulation results

We subject our protocol to extensive experiments by simulation to study its behavior under different network conditions and protocol parameter values. The experiments are intended to study the following metrics and their response to changes in the mobility of nodes, load in the network, node density, and terrain dimensions.

- *Average call acceptance rate*: This metric measures the ability of the protocol to setup the desired number of paths for reliable communication. It depends on both the network topology and the ability of the protocol to successfully make use of available network resources

$$\text{ACAR} = \frac{\text{Number of calls successfully setup}}{\text{Total number of call requests}}. \tag{8}$$

- *Average information delivery ratio*: This ratio indicates the ability of the reliable paths setup to counter adversary events. It is measured as a ratio of the number of data packets received at the destination to the total number of data packets sent by the source. Since schemes use various forms of coding to provide reliability, the number of packets sent by the source refers to the number of unencoded (plain data) packets that have been transmitted. Similarly, the number of data packets received refers to the number of original data packets recovered from the packets received at the destination. For a scheme with $k = 2$ and $b = 2$ (i.e., one in which two packets are encoded as four packets such that the original two can be recovered as long as at most 2 packets out of the 4 are lost), arrival at the destination of 2 packets out of the 4 packets sent implies that the destination can recover the original 2 packets sent. The IDR in this case is $\frac{2}{2} = 1$. On the other hand, arrival of only 1 packet is, in general, equivalent to the loss of the 2 packets transmitted. This is because receipt of a single packet is not enough to recover the 2 packets that were sent. The IDR now is $\frac{0}{2} = 0$. If the encoding happens to be such that the original packets are sent as such and only the additional packets are encoded, then the receiver can still recover some data from a single packet if it happens to be the packet sent unencoded, however this does not hold in the general case

$$\mathrm{IDR} = \frac{\text{Number of data packets successfully delivered}}{\text{Total number of data packets sent by the source}}. \tag{9}$$

- *Resource consumption ratio*: This ratio is a measure of the amount of network resources utilized in sustaining data transmission across the reliable channel established. We directly measure this ratio in terms of the number of packets since all the packets have the same size for CBR traffic. For a given session, its average hopcount is the average over all the paths setup between source and destination at every instant of time

$$\mathrm{RCR} = \frac{\text{Number of data packets transmitted by nodes across the network}}{\text{Total number of data packets sent by the source} \times \text{Average Hopcount}}. \tag{10}$$

- *Control overhead ratio*: This ratio measures the total amount of control data generated in the process of establishing and maintaining the reliable connection. It is measured as a ratio of the number of control bytes generated by nodes across the network to the total number of data bytes sent by the source.

$$\mathrm{COR} = \frac{\text{Number of control bytes transmitted across the network}}{\text{Total number of data bytes sent by the source}}. \tag{11}$$

- *Eavesdropping ratio*: This ratio is a measure of the exposure of the session to nodes other than the source and the destination. For an (n, k)-protocol between two nodes, this is the ratio of the number of nodes that can listen to packets being transmitted on at least k of the paths to the number of nodes that can listen to the packets on a shortest uni-path route between the nodes.

$$\mathrm{ER} = \frac{\text{Number of nodes that can listen to data packets on at least k paths}}{\text{Number of nodes that can listen to the data packets on the shortest uni-path}}. \tag{12}$$

6.1. Simulation results

We simulated our protocol using Glomosim [33]. The network contains 75 nodes in a 1000 m × 1000 m terrain area. The channel capacity is fixed at 2 Mbps and the duration of the simulation is 10 minutes. The transmission range is 282 meters. The traffic is generated in the form of Constant Bit Rate (CBR) sessions each of which lasts for 250 s. The CBR sources generate fixed size packets (512 bytes) at a constant interval of time (500 ms) and do so irrespective of the state of the network.

The load on the network is varied by varying the number of CBR sessions. Mobility is simulated according to the Random waypoint model. According to this model, the nodes remain stationary for a certain pause time after which they move in a randomly chosen direction with a random velocity chosen uniformly between a specified minimum and maximum velocities. For all cases of mobility in the network, we set the pause time to 0 and set the minimum and maximum speeds to the same value to ensure that the nodes move at a constant speed. Most of the simulation has been done assuming that the protocol parameters are known a priori and are fixed for the duration of the simulation for all sessions. The results in Section 6.1.11 include the translation mechanism. As a comparison, the cases where the parameters $n = 1$ or $k = 1$, $b = 0$ correspond to the *DSR protocol*. Results presented in this paper conform to 95% confidence levels.

6.1.1. Average call acceptance rate

We measure the call acceptance rate of the protocol for different values of load on the network at different values of the parameter n. Load is defined as the number of active CBR sessions in the network. Here all nodes are stationary. From Fig. 5, we see that the call acceptance rate decreases with increasing n. The call acceptance rate is determined by two factors: the availability of the required number of node-disjoint paths and the availability of resources along the paths. In this case, with an increase in n, majority of the calls are rejected due to lack of n node-disjoint paths between the source and the destination. The call acceptance rate decreases with an increase in the load, yet the decrease is not considerable showing that the protocol can handle high loads. Fig. 6 shows the variation in the call acceptance rate as the eavesdropping ratio (ER) is varied. A lower ER requires a higher number of paths to be setup, and this reduces the acceptance rate.

6.1.2. Eavesdropping ratio

The eavesdropping ratio was initially measured as a function of k as shown in Fig. 7. For any routing scheme, all the neighbors of the source and some of the neighbors of the destination can listen to all the packets in a batch of n packets. Thus, the eavesdropping ratio is bounded below by the number of neighbors of the source. Fig. 7 shows that as k increases, ER decreases suggesting that a higher value of k reduces the number of eavesdropping nodes. The graph in Fig. 8 shows us the relationship between ER, k, and b. For a fixed value of b, ER decreases with increasing k. On the other hand, as b increases for a fixed k, the number of nodes that can potentially eavesdrop increases. This increases the value of ER. Thus, choosing a value of k of at least two can ensure that the eavesdropping ratio is improved over the unipath routing. For higher values of b, choosing a higher value of k can ensure reliability while enhancing the security.

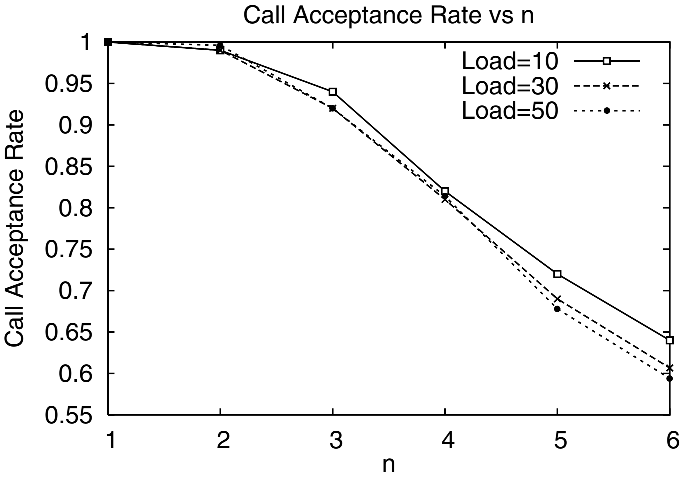


Fig. 5. Variation of call acceptance rate vs n (total number of paths setup) for varying load.

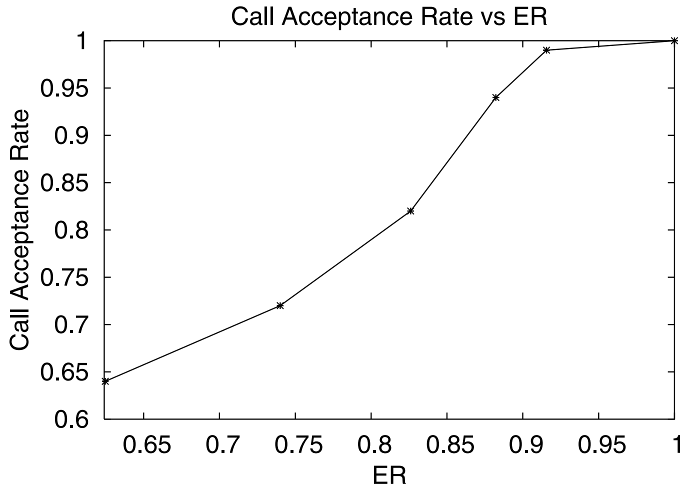


Fig. 6. Variation of call acceptance rate vs eavesdropping ratio (load = 10).

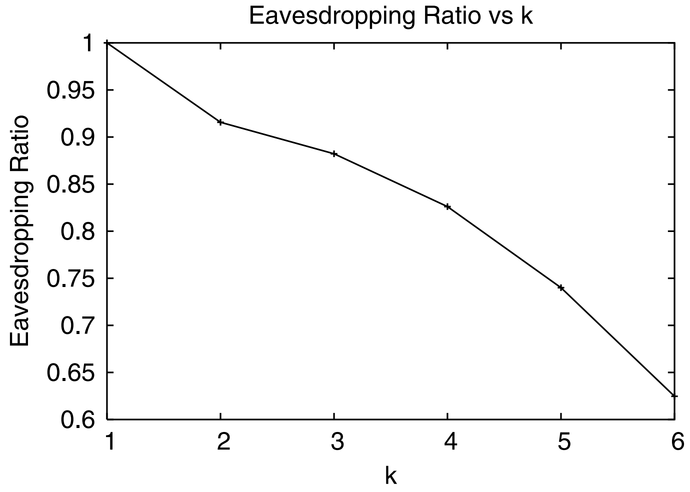


Fig. 7. Variation of eavesdropping ratio vs k ($n = k$, $b = 0$, load = 10).

6.1.3. Information delivery ratio

We have plotted the information delivery ratio for different node velocity values for different values of b in Fig. 9. The value of k has been kept fixed at 2. We have used 10 CBR sessions randomly generated for this experiment. As the velocity increases, the information delivery ratio drops. At low mobility values, the information delivery ratios for various b are nearly identical. However, the higher fault-protection schemes perform better at higher values of mobility. We also study the effect of k and b on the information delivery ratio, Fig. 10, at a fixed mobility value of 12 m/s. For all such graphs in this work, we constrain $k + b = n$ to at most 5 since the call acceptance is too low for higher values of n. While higher values of b increase the information delivery ratio, higher values of k decrease the information delivery ratio. The decrease in IDR

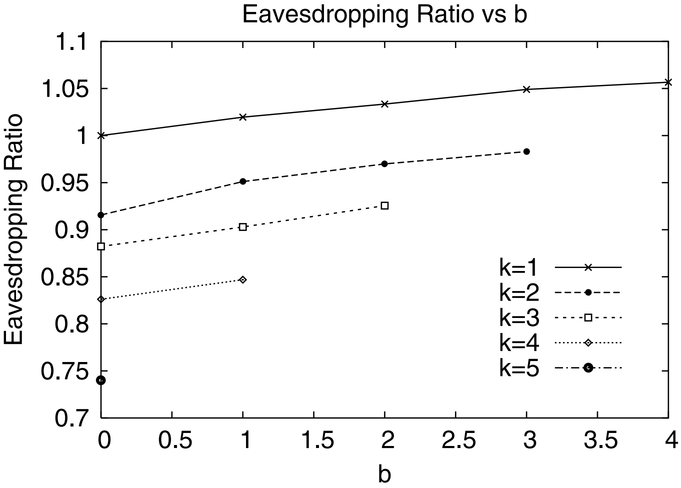


Fig. 8. Variation of eavesdropping ratio vs protocol parameters k and b (load = 10).

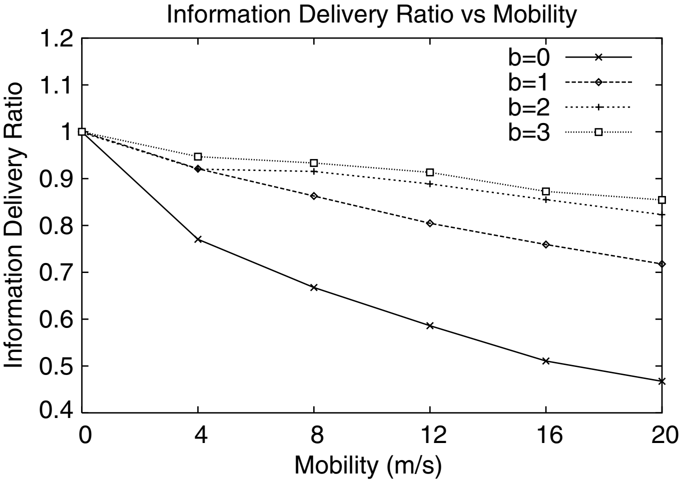


Fig. 9. Variation of information delivery ratio vs mobility for various b ($k = 2$).

due to increasing k can be explained from the fact that with an increase in k, a higher fraction of packets in a batch must reach the destination for the original data packets to be successfully recovered.

6.1.4. Resource consumption ratio

To see the effect of dispersity on the resource consumption, we have plotted the variation of this metric with node velocity for different values of k in Fig. 11. We have kept b fixed at 2 for this comparison. We have used 10 CBR sessions randomly generated for this experiment. With increasing mobility, less number of data packets are transmitted by the network while the number of packets sent remains constant due to the use of a CBR source. By increasing k, we find that the resource consumption ratio decreases as the number of additional paths setup b is constant for a given failure model. At high values of mobility, a decrease in the resource consumption ratio is more likely a result of loss of data packets. We have also plotted the

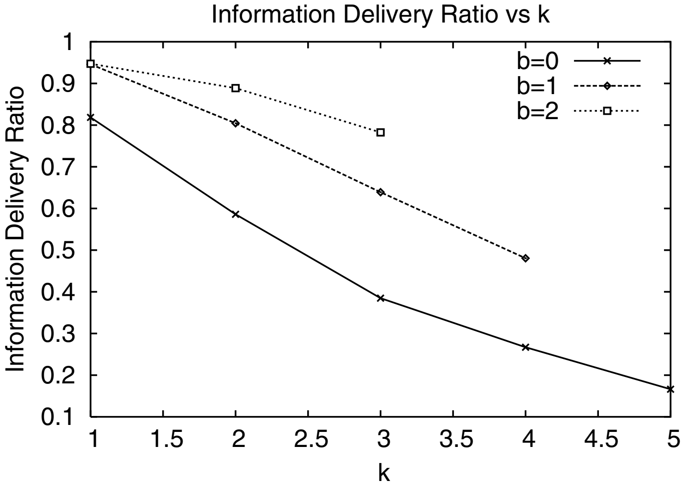


Fig. 10. Variation of information delivery ratio vs protocol parameters k and b (mobility = 12 m/s).

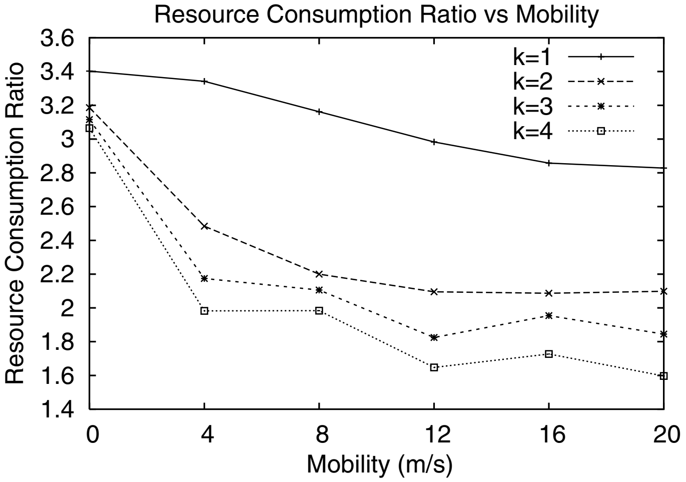


Fig. 11. Variation of resource consumption ratio vs mobility for various k ($b = 2$).

variation of the resource consumption ratio with k and b at a fixed mobility value of 12 m/s in Fig. 12. The values indicate an increase in this ratio with an increase in b due to an increase in the redundancy while a decrease in this ratio occurs with an increase in k since the redundancy is amortized over the increased number of paths.

6.1.5. Control overhead ratio

We have plotted the control overhead ratio for different node velocity values for different values of k in Fig. 13. As we increase the mobility, paths are broken more frequently for the on-going sessions. In DSR protocol, the source node makes use of one of the paths available in its route cache to reconfigure the session on hearing a RERROR packet. However, in MuSeQoR protocol, since the source node only maintains the set of active routes of the session in its S-CACHE, on hearing a RERROR packet the source node ini-

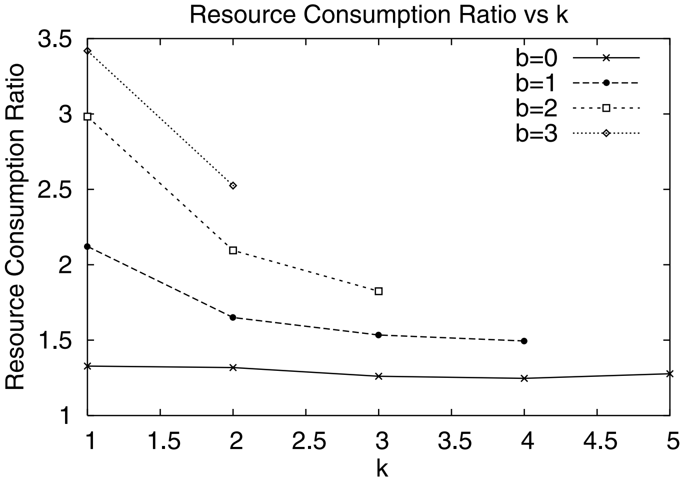


Fig. 12. Variation of resource consumption ratio vs protocol parameters k and b (mobility = 12 m/s).

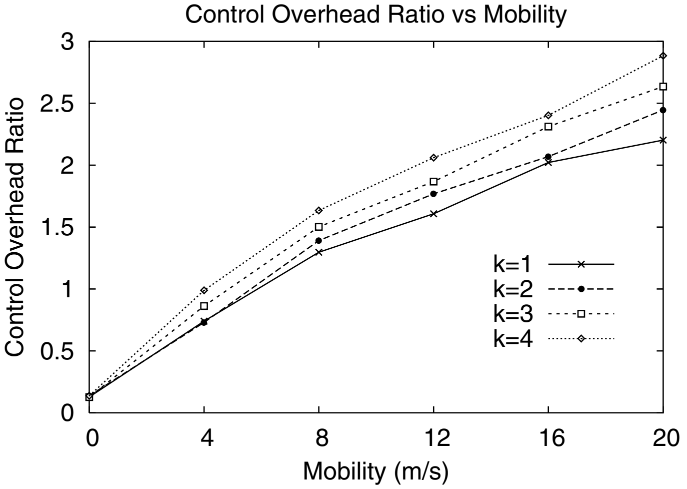


Fig. 13. Variation of control overhead vs mobility for various k ($b = 2$).

tiates another REQUEST to the destination node. Therefore by increasing the mobility in the network, the control overhead increases as more number of control packets are transmitted in order to repair paths broken. As expected the control overhead is higher as a greater number of paths are setup as can be seen from Figs. 13 and 14.

6.1.6. Effect of load

Figs. 15 and 16 show the variation in information delivery ratio and control overhead ratio with changing load for a stationary network. The load was varied by changing the number of sessions across the values 10, 30, and 50. IDR is seen to decrease with an increase in the load. This is because, at higher loads some of the packets get dropped during the transmission due to congestion in the network. The control overhead

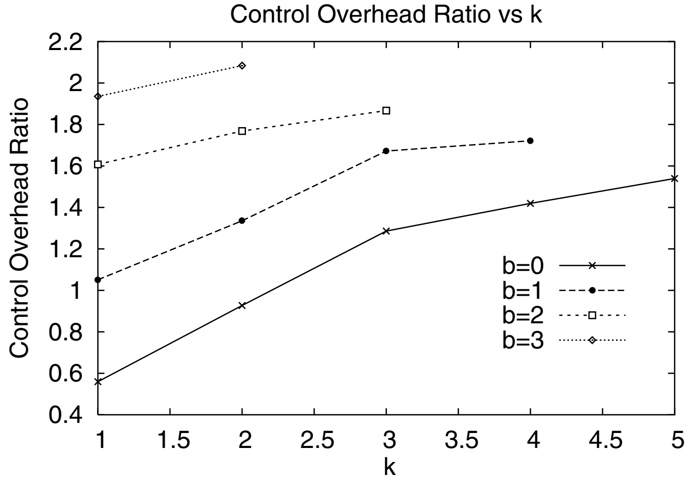


Fig. 14. Variation of control overhead vs protocol parameters *k* and *b* (mobility = 12 m/s).

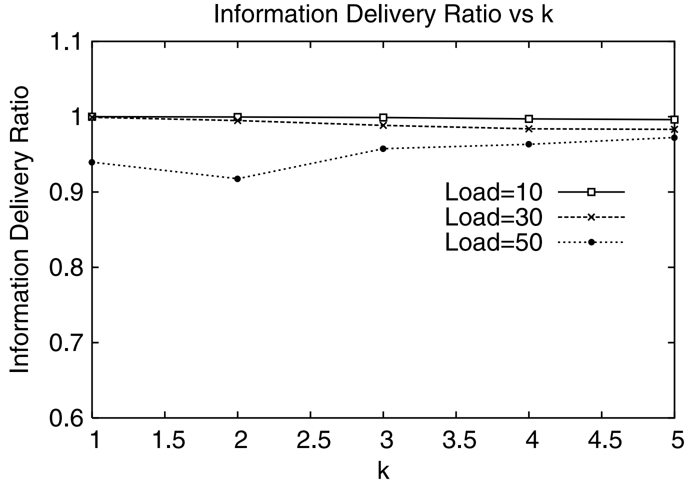


Fig. 15. Variation of information delivery ratio vs *k* ($b = 0$).

ratio increases with an increase in *k* due to the setting up of higher number of paths. COR is seen to increase with an increase in the load. This is because, at higher loads contention for the channel increases. Since REQUEST packets are broadcasted whenever the channel is sensed idle by the MAC protocol (We used the IEEE 802.11 as the underlying MAC protocol which does not use the RTS-CTS control packet exchange to avoid collisions before sending REQUEST packets), with increase in the load less number of REQUEST packets reach the destination node. Due to this, chances of finding enough number of node-disjoint paths decreases as the offered load increases in the network. Hence the source node sends REQUEST packets several times for establishing the session, there by results in increasing the control overhead ratio.

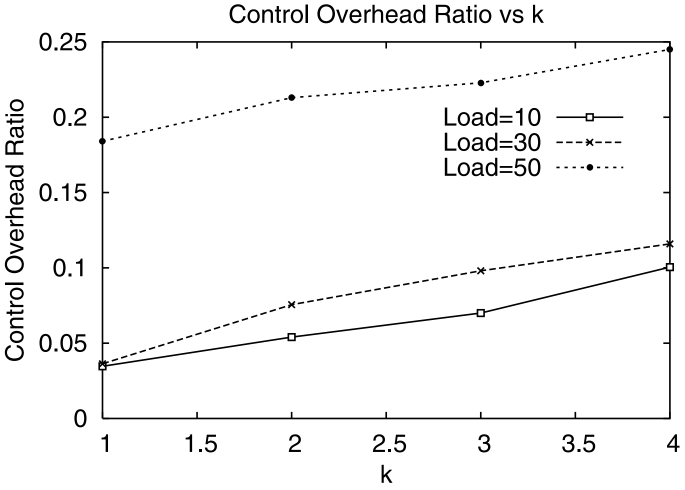


Fig. 16. Variation of control overhead ratio vs k ($b = 0$).

6.1.7. *Effect of terrain dimensions*

We have studied the variation in the call acceptance rate (Fig. 17) and the information delivery ratio (Fig. 18) for different terrain lengths. The terrain size was varied from (500 m × 500 m) to (1500 m × 1500 m) in steps of 250 m. We also consider the cases of mobile nodes moving at a speed of 12 m/s and stationary nodes. The protocol parameters were fixed at $k = 2$ and $b = 1$. We observe that with an increase in the length of the terrain, the call acceptance rate decreases due to the creation of more bottleneck nodes that prevent the setup of node-disjoint paths. However, the call acceptance increases with mobility. This result can be explained by the fact that mobility causes the network to move out of a bottleneck configuration. There is an increased likelihood that some nodes may move into regions where, previously, there were too few nodes to permit node-disjoint paths to be setup. With the source making multiple attempts to setup the paths before rejecting the call, there is now a higher chance for the call to be accepted.

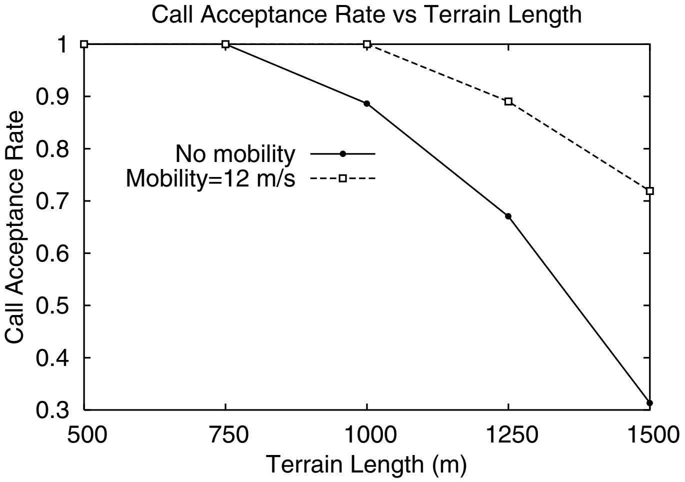


Fig. 17. Variation of call acceptance rate vs terrain dimensions ($k = 2$, $b = 1$, load = 10).

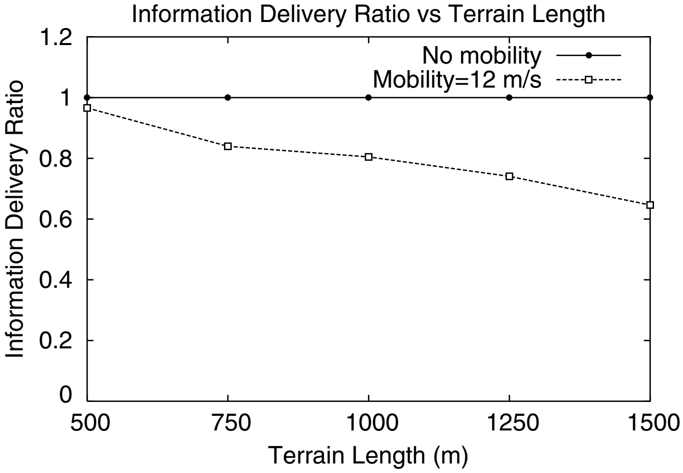


Fig. 18. Variation of information delivery ratio vs terrain dimensions ($k = 2$, $b = 1$, load = 10).

However, mobility lowers the information delivery ratio. The ratio decreases with an increase in the terrain length. The presence of a greater number of hops in the path as a result of an increase in terrain length implies greater chances of packet loss. However for the stationary case, the information delivery ratio remains consistently high.

6.1.8. Effect of node density

In Figs. 19 and 20, the call acceptance rate and the information delivery ratio are measured for varying number of nodes keeping the terrain dimensions fixed. The call acceptance rate is again higher for the mobile case. The acceptance rate increases with an increase in the number of nodes. This is due to an increase in the number of control packets (REQUESTs) being forwarded since a larger number of nodes are present in the transmission range of each node (Fig. 21). As a result, the number of REQUEST packets reaching

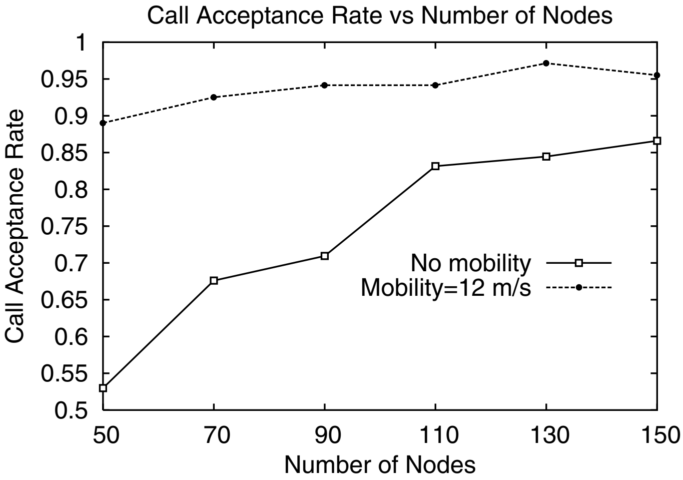


Fig. 19. Variation of call acceptance rate vs number of nodes ($k = 2$, $b = 2$, load = 10).

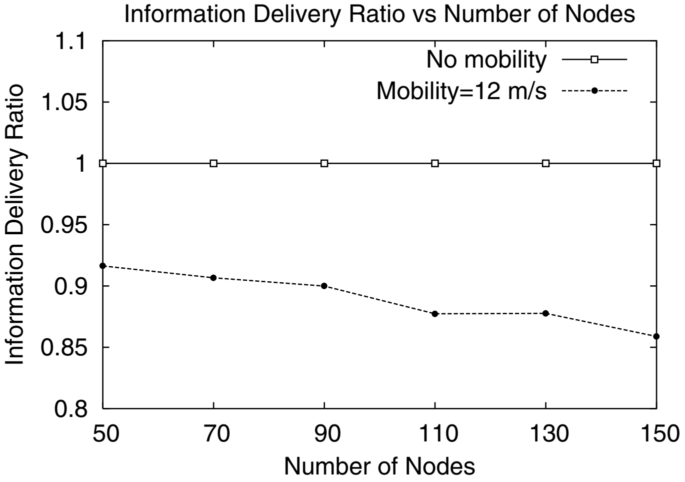


Fig. 20. Variation of information delivery ratio vs number of nodes ($k = 2$, $b = 2$, load = 10).

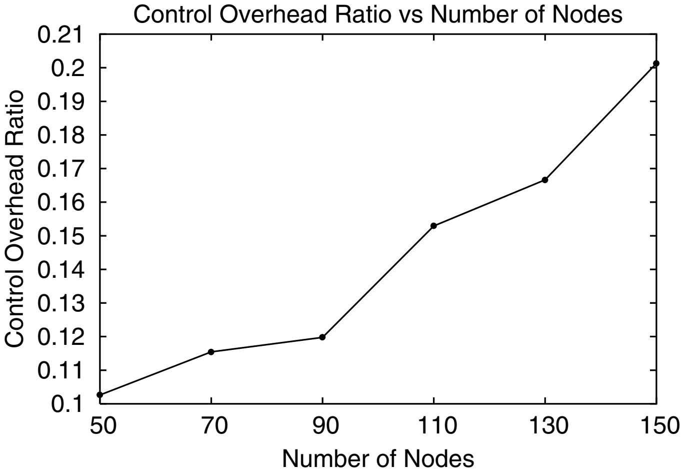


Fig. 21. Variation of control overhead ratio vs number of nodes for a stationary network ($k = 2$, $b = 2$, load = 10).

the destination is greatly increased and the destination may be able to find the required number of node-disjoint paths easily. The information delivery ratio remains fairly constant for both the stationary and the mobile cases while it is slightly lower for the mobile case.

6.1.9. Overheads involved in corruption channels

Table 3 shows the resource consumption ratio and the control overhead ratio for setting up reliable paths in the case of corruption channels. The case of $g = 0$ corresponds to the case where the channel is modeled as an erasure channel. The statistics have been obtained for the case of $k = 2$ and mobility = 12 m/s. It is evident from Table 3 that the RCR and the COR increase as we increase the number of corruptions to be protected against. Also, the increase is much higher for a higher value of f (the number of path failures or erasures that we are protecting against).

Table 3
The RCR and the COR involved in setting up reliable sessions in the presence of corruption channels for $k = 2$ and mobility = 12 m/s

f	*g*	RCR	COR	*f*	*g*	RCR	COR
0	0	0.793176	0.04757	2	0	1.58341	0.10164
0	1	1.58341	0.10164	2	1	2.37503	1.28810
0	2	2.37503	1.28810				
1	0	1.21824	0.11218	3	0	2.06826	0.67117
1	1	2.06826	0.67117	3	1	2.79729	0.91877
1	2	2.79729	0.91877				

6.1.10. Comparison of block allocation strategies

In our protocol, the packets are uniformly distributed over the paths setup. In [18], allocation of packets to the paths is done so that the probability of successful reception is maximized. The success of the latter scheme crucially depends on the estimation of the individual path reliabilities. Path reliability estimation that does not take into account the load on the path may lead to heavy loading of certain paths. Also, transmission of a majority of packets of a batch on a single path allows eavesdropping nodes to listen to a larger fraction of the packets reducing the security of the connection. We have compared the information delivery ratio (IDR) and the eavesdropping ratio (ER) of our protocol using uniform allocation to that which uses non-uniform allocation. The study was done by setting up 6 paths ($k = 3$, $b = f = 3$) between the source and the destination in a static network. The reliability of the paths was simulated by dropping packets at the destination. The numerical figures of the example in Section 3-B of [34] were used. The reliability of all the paths except one were fixed at 0.8 while the remaining path's reliability (given by q) was varied between 0.8 and 1. The two allocation vectors (a tuple that indicates the number of blocks sent on the different paths) compared were $(3,1,1,1,0,0)$ and $(1,1,1,1,1,1)$ i.e., for the first allocation vector, 3 blocks are sent on the first path and 1 block each on the second, third, and fourth paths; for the second allocation vector, a single block is sent on each path. Figs. 22 and 23 indicate that the two schemes have comparable IDR while uniform allocation has a lower ER (lower by about 12%).

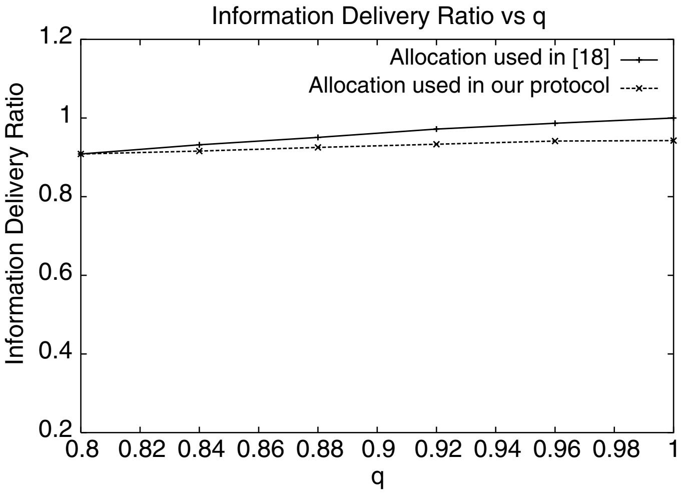


Fig. 22. Comparison of information delivery ratios of the allocation scheme used in [18] and our allocation scheme.

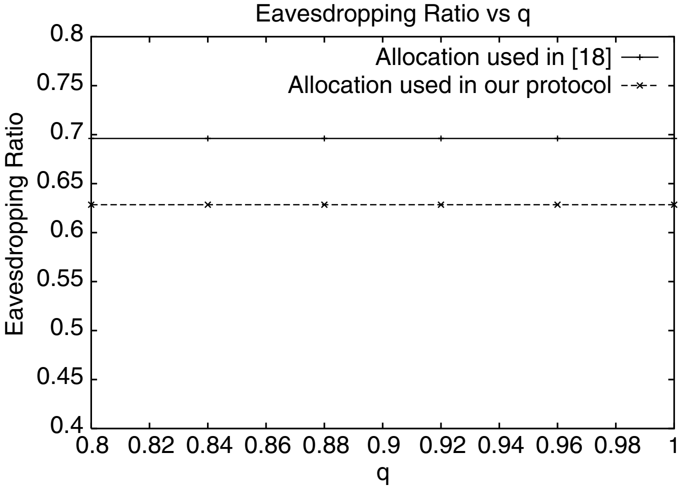


Fig. 23. Comparison of eavesdropping ratios of the allocation scheme used in [18] and our allocation scheme.

6.1.11. Dynamic determination of *f*—a translation mechanism

MuSeQoR is adaptive as it determines the failure model based on the state of the network. This allows the user to specify the QoS requirements in terms of the overall failure probability required which is then converted to the protocol parameter f (which determines the fault model). Such a translation mechanism was implemented to measure the link availability and to determine the value of f. The path-failure metric ϵ was specified as 0.3. The values of f determined by this mechanism were measured at different values of mobility. As the mobility increases, higher values of f are estimated for the fault model (Fig. 24).

The other plot (Fig. 25) shows the impact of the path-failure metric ϵ and how the protocol adapts to this metric. The smaller the value of ϵ, the higher the reliability demanded by the user: since ϵ denotes the threshold failure that can be tolerated by the application. This experiment was done in a high-mobility scenario of

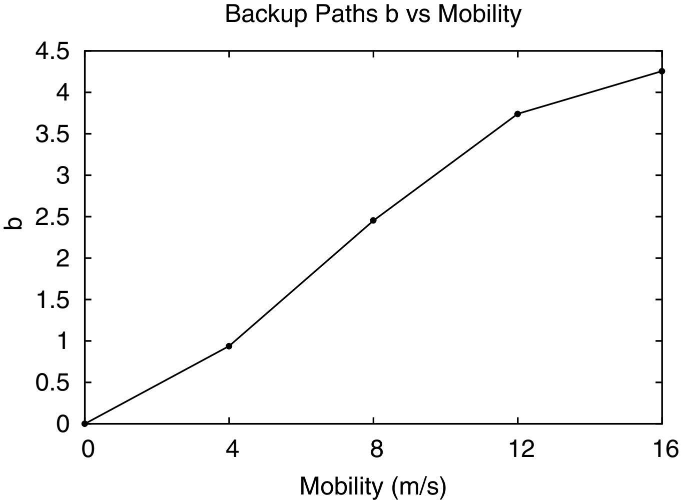


Fig. 24. Variation of b (=f) with mobility when the protocol attempts to adapt to the network.

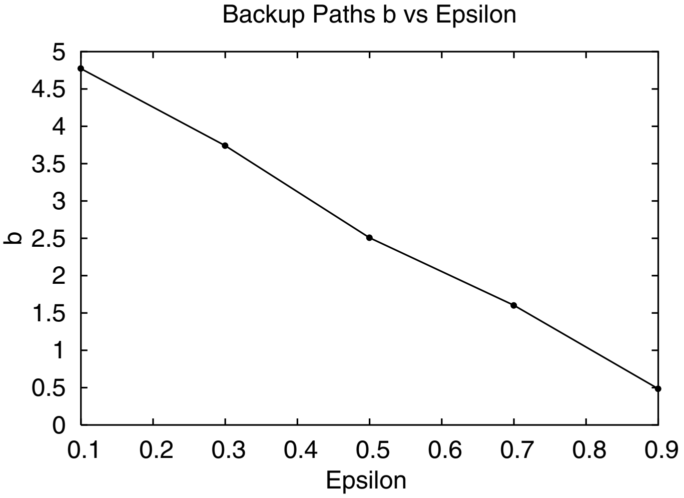


Fig. 25. Variation of b (=f) with the path-failure metric epsilon (ϵ) at a mobility of 12 m/s.

12 m/s. In such a scenario, to satisfy a more stringent (lower) value of ϵ, more paths need to be setup since the paths by themselves do not have a high reliability. This experiment shows how the system adapts to the reliability requirements by setting up more paths.

7. Conclusion and future work

The issues of reliability and security are of growing importance in Ad hoc wireless networks. MuSeQoR attempts to address these two issues while ensuring that the data overhead involved is minimum (since the coding schemes used are optimal). The protocol is characterized by the parameters k, f, and g. k represents the number of blocks into which a packet is divided before coding. f is the maximum number of path-failures that can be tolerated by the protocol and g is the maximum number of adversary nodes to be protected against. We related k and f to the user requirements specified by ϵ: the path failure metric and ER: the eavesdropping ratio. A higher value of f indicates higher reliability while a higher value of k implies a lower eavesdropping ratio and a lower redundancy ratio. The parameters k and f are also constrained by the network topology.

The information delivery ratios are higher for high values of b while the redundancy ratio and the eavesdropping ratio decrease for high values of k. For the static scenario, values of k and b between 2 and 3 offer a good compromise. In all the cases, the advantages over uni-path routing are evident. We also demonstrated the adaptability of the protocol for a mobile scenario and a given reliability metric. Finally, we compared the block allocation strategy of [18] with the allocation strategy used in our scheme, which allocates packets equally on each path. We then studied the effect of the translation mechanism. The adaptability of the protocol is seen from the number of additional paths setup in an increasingly mobile scenario.

Further work needs to be done to answer the following issues: What would be the nature of the QoS guarantees that can be provided if we relax the condition of node-disjointedness of the paths? How do we build protocols that tackle multiple QoS parameters so that they can respond to the needs of multiple classes of users? We need to explore the interaction of other potential applications that require the security and reliability guarantees that MuSeQoR provides with the routing protocol. These include multimedia streaming and mobile hospitals among others. We also would like to see how the scheme performs in other local broadcast networks such as sensor networks which have a small variation in the constraints.

References

[1] V. Marbukh, A cognitive framework for performance/resilience optimized multi-path routing in networks with unstable topologies, in: Proceedings of IEEE WCNC 2003, vol. 2, March 2003, pp. 1149–1154.

[2] A. Nasipuri, S.R. Das, On-demand multi-path routing for mobile ad hoc networks, in: Proceedings of IEEE ICCCN 1999, October 1999, pp. 64–70.

[3] G. Xue, Optimal multi-path end-to-end data transmission in networks, in: Proceedings of IEEE ISCC 2000, July 2000, pp. 581–586.

[4] K. Wu, J. Harms, On-demand multi-path routing for mobile ad hoc networks, in: Proceedings of 4th European Personal Mobile Communication Conference (EPMCC 2001), CD-ROM Proceedings, February 2001, Paper no. 21.1.

[5] Y. Zhong, X. Yuan, Impact of resource reservation on the distributed multi-path quality of service routing scheme, in: Proceedings of IEEE IWQoS 2000, June 2000, pp. 95–104.

[6] L. Wang, Y. Shu, M. Dong, L. Zhang, O.W.W. Yang, Adaptive multi-path source routing in ad hoc networks, in: Proceedings of IEEE ICC 2001, vol. 3, June 2001, pp. 867–871.

[7] L. Wang, L. Zhang, Y. Shu, M. Dong, Multi-path source routing in wireless ad hoc networks, in: Proceedings of 2000 Canadian Conference on Electrical and Computer Engineering, vol. 1, pp. May 2000, 479–483.

[8] S. Lee, M. Gerla, Split multi-path routing with maximally disjoint paths in ad hoc networks, in: Proceedings of IEEE ICC 2001, vol. 10, June 2001, pp. 3201–3205.

[9] S. De, S.K Das, Dynamic multi-path routing (DMPR): an approach to improve resource utilization in networks for real-time traffic, in: Proceedings of IEEE MASCOTS 2001, August 2001, pp. 23–32.

[10] C.K. Siew, G. Wu, G. Feng, On-demand QoS multi-path routing, in: Proceedings of IEEE ICCS 2003, vol. 1, November 2002, pp. 589–593.

[11] V. Srinivas, J.J. Garcia-Luna-Aceves, MDVA: a distance-vector multi-path routing protocol, in: Proceedings of IEEE INFOCOM 2001, vol. 1, April 2001, pp. 557–564.

[12] M.K. Marina, S.R. Das, On-demand multi-path distance-vector routing in ad hoc networks, in: Proceedings of IEEE ICNP 2001, November 2001, pp. 14–23.

[13] X. Lin, I. Stojmenovic, Location-based localized alternate, disjoint and multi-path routing algorithms for wireless networks, J. Parallel Distrib. Comput. 63 (1) (2003) 22–32.

[14] Z. Yao, Z. Ma, Z. Cao, A multi-path routing scheme combating with frequent topology changes in wireless ad hoc networks, in: Proceedings of ICCT 2003, vol. 2, April 2003, pp. 1250–1253.

[15] A. Valera, W.K.G. Seah, S.V. Rao, Cooperative packet caching and shortest multi-path routing in mobile ad hoc networks, in: Proceedings of IEEE INFOCOM 2003, vol. 1, April 2003, pp. 260–269.

[16] R. Ma, J. Ilow, Reliable multi-path routing with fixed delays in MANET using regenerating nodes, in: Proceedings of IEEE LCN 2003, October 2003, pp. 719–725.

[17] E. Ayanoglu, I. Chih-Lin, R.D. Gitlin, J.E. Mazo, Diversity coding for transparent self-healing and fault-tolerant communication networks, IEEE Trans. Commun. 41 (11) (1993) 1677–1686.

[18] A. Tsirigos, Z.J. Haas, Multi-path routing in the presence of frequent topological changes, IEEE Commun. Mag. 39 (11) (2001) 132–138.

[19] R. Leung, J. Liu, E. Poon, A.C. Chan, B. Li, MP-DSR: a QoS-aware multi-path dynamic source routing protocol for wireless ad hoc networks, in: Proceedings of IEEE LCN 2001, November 2001, pp. 132–141.

[20] L. Chou, C. Hsu, F. Wu, A reliable multi-path routing protocol for ad hoc network, in: Proceeding of IEEE ICON 2002, August 2002, pp. 305–310.

[21] W. Lou, Y. Fang, A multi-path routing approach for secure data delivery, in: Proceedings of IEEE MILCOM 2001, vol. 2, October 2001, pp. 1467–1473.

[22] C.K. Lee, X. Lin, Y. Kwok, A multi-path ad hoc routing approach to combat wireless link insecurity, in: Proceedings of IEEE ICC 2003, 2003, vol. 1, pp. 448–452.

[23] D.B. Johnson, D.A. Maltz, in: Dynamic Source Routing in Ad hoc Wireless NetworksMobile Computing, vol. 353, Kluwer Academic Publishers, 1996, pp. 153–181.

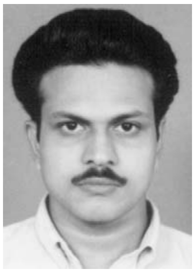
[24] A. Nasipuri, R. Castaneda, S.R. Das, Performance of multi-path routing for on-demand protocols in mobile ad hoc networks, ACM/Baltzer Mobile Networks Appl. (MONET) J. 6 (4) (2001) 339–349.

[25] S.K. Das, A. Mukherjee, S. Bandyopadhyay, K. Paul, D. Saha, Improving quality of service in Ad hoc wireless networks with adaptive multi-path routing, in: Proceedings of IEEE GLOBECOM 2000, vol. 1, December 2000, pp. 261–265.

[26] Y. Chen, R. Hwang, Y. Lin, Multi-path QoS routing with bandwidth guarantee, in: Proceedings of IEEE GLOBECOM 2001, vol. 4, November 2001, pp. 2199–2203.
[27] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, IETF RFC2475, December 1998.
[28] S.S. Lee, S. Das, G. Pau, M. Gerla, A hierarchical multi-path approach to QoS routing: performance and cost evaluation, in: Proceedings of IEEE ICC 2003, 2003, vol. 1, pp. 625–630.
[29] A.B. McDonald, T. Znati, A path availability model for wireless ad hoc networks, in: Proceedings of IEEE WCNC 1999, vol. 1, September 1999, pp. 35–40.
[30] R. Lidl, H. Niederreiter, Finite Fields, Cambridge University Press, London, 1997.
[31] J. Gallian, Contemporary Abstract Algebra, Houghton Mifflin, Boston, 2000.
[32] R.E. Blahut, Algebraic Coding for Data Transmission, Cambridge University Press, London, 2002.
[33] X. Zheng, R. Bagrodia, M. Gerla, GloMoSim: a library for parallel simulation of large-scale wireless networks, in: Proceedings of the 12th Workshop on Parallel and Distributed Simulations (PADS 1998), May 1998, pp. 154–161.
[34] A. Tsirigos, Z.J. Haas, Analysis of multi-path routing—Part I: The effect on the packet delivery ratio, IEEE Trans. Wireless Commun. 3 (1) (2004) 138–146.

Tamma Bheemarjuna Reddy received the B.Tech. degree in Computer Science and Engineering from Andhra University, India, in 2000 and the M.E. degree in Computer Science and Engineering from the National Institute of Technology (NIT), Rourkela, India, in 2002. He is currently a doctoral student in the Department of Computer Science and Engineering at the Indian Institute of Technology (IIT), Madras, India. His research interests include QoS provisioning and Multimedia transport in Adhoc wireless networks.

S. Sriram obtained his B.Tech. degree in Computer Science and Engineering in 2004 from the Indian Institute of Technology (IIT), Madras, India. He is currently working towards the Ph.D. degree in the department of Computer Science at the University of California, Berkeley, USA. His research interests include wireless networks, distributed systems, network security, and computational biology.



B.S. Manoj completed his graduation in 1995 and post graduation in 1998 both in Electronics and Communication Engineering from Institution of Engineers (India) and Pondicherry Central University, Pondicherry, India, respectively. He has worked as a Senior Engineer with Banyan Networks Pvt. Ltd., Chennai, India from 1998 to 2000 where his primary responsibility included design and development of protocols for real-time traffic support in data networks. He was an Infosys doctoral student during 2000-2003 in the Department of Computer Science and Engineering at the Indian Institute of Technology (IIT) Madras, India, where he focused on the development of architectures and protocols for Ad hoc wireless networks and next generation hybrid wireless network architectures. During January 2004–January 2005, he was a Project Officer at IIT, Madras, India. He is currently a post-doctoral researcher at the University of California, San Diego, USA. Indian Science Congress Association has awarded him the Young Scientist Award for the Year 2003. His current research interests include Ad hoc wireless networks, next generation wireless architectures, and wireless sensor networks.

C. Siva Ram Murthy received the B.Tech. degree in Electronics and Communications Engineering from Regional Engineering College (now National Institute of Technology), Warangal, India, in 1982, the M.Tech. degree in Computer Engineering from the Indian Institute of Technology (IIT), Kharagpur, India, in 1984, and the Ph.D. degree in Computer Science from the Indian Institute of Science, Bangalore, India, in 1988.

He joined the Department of Computer Science and Engineering at IIT, Madras, as a Lecturer in September 1988, and became an Assistant Professor in August 1989 and an Associate Professor in May 1995. He has been a Professor with the same department since September 2000. He has held visiting positions at the German National Research Centre for Information Technology (GMD), Bonn, Germany, the University of Stuttgart, Germany, the University of Freiburg, Germany, the Swiss Federal Institute of Technology (EPFL), Switzerland, and the University of Washington, Seattle, USA.

He is the co-author of the textbooks Parallel Computers: Architecture and Programming, (Prentice-Hall of India, New Delhi, India), New Parallel Algorithms for Direct Solution of Linear Equations, (John Wiley & Sons, Inc., New York, USA), Resource Management in Real-time Systems and Networks, (MIT Press, Cambridge, Massachusetts, USA), WDM Optical Networks: Concepts, Design, and Algorithms, (Prentice Hall, Upper Saddle River, New Jersey, USA), and Ad Hoc Wireless Networks: Architectures and Protocols, (Prentice Hall, Upper Saddle River, New Jersey, USA). His research interests include parallel and distributed computing, real-time systems, lightwave networks, and wireless networks. He has published more than 200 technical papers in these areas.

He is a recipient of the Sheshgiri Kaikini Medal for the Best Ph.D. Thesis from the Indian Institute of Science, the Indian National Science Academy (INSA) Medal for Young Scientists, and Dr. Vikram Sarabhai Research Award for his scientific contributions and achievements in the fields of Electronics, Informatics, Telematics, and Automation. He is a co-recipient of Best Paper Awards from the 1st Inter Research Institute Student Seminar (IRISS) in Computer Science, the 5th IEEE International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), and the 6th and 11th International Conference on High Performance Computing (HiPC).