An Iterative Algorithm for Trust Management and Adversary Detection for Delay-Tolerant Networks

Erman Ayday, *Student Member*, *IEEE*, and Faramarz Fekri, *Senior Member*, *IEEE*

Abstract—Delay/Disruption Tolerant Networks (DTNs) have been identified as one of the key areas in the field of wireless communication, wherein sparseness and delay are particularly high. They are emerging as a promising technology in vehicular, planetary/interplanetary, military/tactical, disaster response, underwater and satellite networks. DTNs are characterized by large end-to-end communication latency and the lack of end-to-end path from a source to its destination. These characteristics pose several challenges to the security of DTNs. Especially, Byzantine attacks in which one or more legitimate nodes have been compromised and fully controlled by the adversary can give serious damages to the network in terms of latency and data availability. Using reputation-based trust management systems is shown to be an effective way to handle the adversarial behavior in Mobile Ad hoc Networks (MANETs). However, because of the unique characteristics of DTNs, those traditional techniques do not apply to DTNs. Our main objective in this paper is to develop a robust trust mechanism and an efficient and low cost malicious node detection technique for DTNs. Inspired by our recent results on reputation management for online systems and e-commerce, we develop an iterative malicious node detection mechanism for DTNs referred as ITRM. The proposed scheme is a graph-based iterative algorithm motivated by the prior success of message passing techniques for decoding low-density parity-check codes over bipartite graphs. Applying ITRM to DTNs for various mobility models, we observed that the proposed iterative reputation management scheme is far more effective than well-known reputation management techniques such as the Bayesian framework and EigenTrust. Further, we concluded that the proposed scheme provides high data availability and packet-delivery ratio with low latency in DTNs under various adversary attacks which attempt to both undermine the trust and detection scheme and the packet delivery protocol.

Index Terms—Security, trust and reputation management, iterative algorithms, malicious node detection, delay-tolerant networks.

1 Introduction

Delay-tolerant Networks (henceforth referred to as DTNs) are a relatively new class of networks [1], wherein sparseness and delay are particularly high. In conventional Mobile Ad hoc Networks (MANETs), the existence of end-to-end paths via contemporaneous links is assumed in spite of node mobility. It is also assumed that if a path is disrupted due to mobility, the disruption is temporary and either the same path or an alternative one is restored very quickly. In contrast, DTNs are characterized by intermittent contacts between nodes, leading to space-time evolution of multihop paths (routes) for transmitting packets to the destination. In other words, DTNs' links on an end-to-end path do not exist contemporaneously, and hence intermediate nodes may need to store, carry, and wait for opportunities to transfer data packets toward their destinations. Hence, DTNs are much more general than MANETs in the mobile network space (i.e., MANETs are special types of DTNs). Applications of DTNs include emergency response, wildlife surveying, vehicular-to-vehicular communications, healthcare, military, and tactical sensing.

• *The authors are with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332. E-mail: eayday@gatech.edu, fekri@ece.gatech.edu.*

Manuscript received 15 Feb. 2010; revised 5 July 2011; accepted 15 July 2011; published online 1 Aug. 2011.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2010-02-0076. Digital Object Identifier no. 10.1109/TMC.2011.160.

Compared to traditional MANETs, common problems in packet communication such as routing, unicasting, broadcasting and multicasting become sufficiently harder in DTNs even with lossless links (i.e., no packet erasures due to communication link). This increase in difficulty can be directly attributed to the lack of knowledge on the network topology, and the lack of end-to-end path. Hence, the schemes for routing packets have to be primitive such as forwarding to the next available node, injecting multiple copies into available nodes and employing erasure block codes [2]. On the other hand, depending upon the model for mobility, efficient communication schemes for stationary ad hoc networks can be extended partially or wholly to DTNs.

As in MANETs, adversary may mount several threats against DTNs to reduce the performance of the network. The most serious attacks are due to the Byzantine (insider) adversary in which one or more legitimate nodes have been compromised and fully controlled by the adversary. A Byzantine-malicious node may mount the following attacks in order to give serious damage to the network:

1. Packet drop, in which the malicious node drops legitimate packets to disrupt data availability,
2. Bogus packet injection, in which the Byzantine node injects bogus packets to consume the limited resources of the network,
3. Noise injection, in which the malicious node changes the integrity of legitimate packets,

4. Routing attacks, in which the adversary tempers with the routing by misleading the nodes,
5. Flooding attacks, in which the adversary keeps the communication channel busy to prevent legitimate traffic from reaching its destination, and
6. Impersonation attacks, in which the adversary impersonates the legitimate nodes to mislead the network.

We note that because of the lack of end-to-end path from a source to its destination in DTNs, routing attacks are not significant threats for such networks. Attacks on packet integrity may be prevented using a robust authentication mechanism in both MANETs and DTNs. However, packet drop is harder to contain because nodes' cooperation is fundamental for the operation of these networks (i.e., a group of nodes cooperate in routing each others' packets using multihop wireless links without any centralized control). This cooperation can be undermined by Byzantine attackers, selfish nodes, or even innocent but faulty nodes. Therefore, in this work, we focus on packet drop attack which gives serious damages to the network in terms of data availability, latency, and throughput. Finally, Byzantine nodes may individually or in collaboration attack the security mechanism (e.g., the trust management and malicious node detection schemes) as will be discussed later.

In MANETs, reputation-based trust management systems are shown to be an effective way to cope with adversary. By establishing trust with the nodes it has or has not directly interacted, a node in the network diagnoses other nodes and predicts their future behavior in the network. Hence, trust plays a pivotal role for a node in choosing with which nodes it should cooperate, improving data availability in the network. Further, examining trust values has been shown to lead to the detection of malicious nodes in MANETs. Despite all the progress for securing MANETs, achieving the same for DTNs leads to additional challenges. The special constraints posed by DTNs make existing security protocols inefficient or impractical in such networks as will be discussed in Section 1.1.

Our main objective in this paper is to develop a security mechanism for DTNs which enables us to evaluate the nodes based on their behavior during their past interactions and to detect misbehavior due to Byzantine adversaries, selfish nodes, and faulty nodes. The resulting scheme would effectively provide high data availability and packet delivery ratio with low latency in DTNs in the presence of Byzantine attackers. To achieve this goal, we aim at obtaining a reputation-based trust management system and an iterative malicious node detection mechanism for DTNs. Our work on reputation systems stems from the prior success of iterative algorithms, such as message passing techniques [3] in the decoding of Low-Density Parity-Check (LDPC) codes in erasure channels [4]. We believe the significant benefits offered by iterative algorithms can be tapped in to benefit the field of reputation systems. To achieve this, we develop the Iterative Trust and Reputation Mechanism (ITRM) [5], and explore its application on DTNs. We propose a distributed malicious node detection mechanism for DTNs using ITRM which enables every node to evaluate other nodes based on their past behavior, without requiring a central authority. We will show that the resulting scheme effectively provides high data availability and low latency in the presence of Byzantine attackers. We will also show that the proposed iterative mechanism is far more effective than the voting-based techniques in detecting Byzantine nodes.

The main contributions of our work are summarized in the following:

1. We introduced a novel iterative method for trust and reputation management referred as ITRM which is inspired by the iterative decoding of low-density parity-check codes over bipartite graphs.
2. We introduce the application of ITRM into DTNs as an iterative trust management and malicious node detection scheme. The scheme provides high data availability and packet delivery ratio with low latency in the presence of Byzantine attackers.
3. The proposed algorithm computes the reputations of the network nodes accurately in a short amount of time in the presence of attackers without any central authority.
4. The proposed algorithm mitigates the impacts of Byzantine attackers proportional to their attack degrees. That is, the ones that are attacking with the highest strength are detected with higher probability.
5. Comparison of ITRM with some well-known reputation management techniques (e.g., *Bayesian framework* and *EigenTrust*) indicates the superiority of ITRM in terms of robustness against attacks in a realistic DTN environment. Further, the proposed algorithm is very efficient in terms of its computational complexity. Specifically, the complexity of ITRM is linear in the number of nodes. Hence, it is scalable and suitable for large-scale implementations.

The rest of this paper is organized as follows: In the rest of this section, we summarize the related work. In Section 2, we describe ITRM and its security evaluation as a general trust and reputation framework (i.e., in a general setting). Next, in Section 3, we present the application of ITRM to DTNs and the proposed security mechanism in detail. Moreover, we evaluate the proposed scheme by analysis and simulations in a realistic DTN environment. Finally, in Section 4, we conclude the paper.

1.1 Related Work

The main goal for building a reputation system in MANETs is to protect the reactive routing protocol from attackers and increase the performance of the network. A recent review of these secure routing protocols for MANETs [6] indicates that these protocols either use the watchdog mechanism or ACK messages to build trust values between the nodes. In MANETs, a node evaluates another by using either direct or indirect measurements. Building reputation values by direct measurement is either achieved by using the watchdog mechanism or by using the ACK from the destination. Building reputation values by just relying on the direct measurements and using the watchdog mechanism is proposed in [7], [8]. In [9], [10], the use of indirect measurements to build reputation values is also allowed

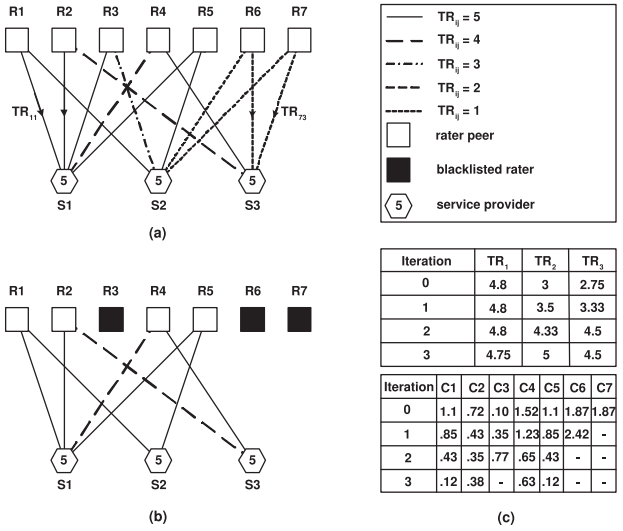
while the watchdog mechanism is used to obtain direct measurements. In [11], [12], [13], [14], [15], reputation values are constructed using the ACK messages sent by the destination node. We note that these techniques are not applicable to DTNs due to the following reasons. In DTNs, a node cannot use the watchdog mechanism and monitor another intermediate node after forwarding its packets to it. This is because links on an end-to-end path do not exist contemporaneously, and hence an intermediate node needs to store, carry, and wait for opportunities to transfer those packets. As a result, the node loses connection with the intermediate node which it desires to monitor. This implies that a Byzantine node in DTNs can get packets from a legitimate node, then move away and drop the packets. Similarly, relying on the ACK packets from the destination to establish reputation values would fail in DTNs because of the lack of a fixed common multihop path from the source to the destination. Even if we assume an ACK from destination to the source (which incurs large latency), this feedback packet travels to the source via intermediate nodes that are different from the set of nodes that delivered the data packet to the destination. More specifically, the source node, upon receiving a negative ACK, cannot decide which node on the forwarding path is to be blamed. Lastly, using indirect measurements is possible in DTNs. However, it is unclear as to how these measurements can be obtained in the first place.

Reputation systems for P2P networks and online systems also received a lot of attention [10], [16], [17], [18], [19], [20], [21], [22]. In [16] and [17], authors cover most of the work on the use of reputation systems for P2P networks. However, reputation systems for P2P networks are either not applicable for DTNs or they require excessive time to build the reputation values of the peers. Most proposed P2P reputation management mechanisms utilize the idea that a peer can monitor others and obtain direct observations [18] or a peer can enquire about the reputation value of another peer (and hence, obtain indirect observations) before using the service provided by that peer [19], [20]. However, neither of these techniques are practical for DTNs. In DTNs, direct observations are not possible as we discussed above. Further, enquiring about the reputation value of a peer is not practical in DTNs due to opportunistic communications during contact times and intermittent connectivity of the peers. Assuming a peer enquires about the reputation values of the other peers from its contacts, it can calculate the reputation values of the other peers when it collects sufficient indirect measurements. However, considering the opportunistic and intermittent connectivity in DTNs, this method requires excessive time to build the reputation values of all peers in the network. EigenTrust [21] is one of the most popular reputation management algorithm for P2P networks. However, the EigenTrust algorithm is constrained by the fact that trustworthiness of a peer (on its feedback) is equivalent to its reputation value. In EigenTrust, the trust relationships between the nodes are established based on the service qualities of the peers during a P2P file transfer. However, trusting a peer's feedback and trusting a peer's service quality are two different concepts. As we will discuss in Section 3.1, a malicious peer can attack the network protocol or the reputation management system independently. Therefore, the EigenTrust algorithm is not practical for applications in which the trustworthiness and reputation are two separate concepts (as in our work). Use of the Bayesian framework is also proposed in [9]. In schemes utilizing the Bayesian framework, each reputation value is computed independent of the other nodes' reputation values. However, the ratings provided by the nodes induce a probability distribution on the reputation values. These distributions are correlated because they are induced by the overlapping set of nodes. The strength of ITRM stems from the fact that it tries to capture this correlation in analyzing the ratings and computing the reputation values. Finally, Dellarocas [22] proposed to use the *Cluster Filtering* method [23] for reputation management. However, it can be shown that Cluster Filtering introduces quadratic complexity while the computational complexity of ITRM is linear with the number of users in the network. As a result, our proposed scheme is more scalable and suitable for large-scale reputation systems. Different from the existing schemes, ITRM algorithm [5] is a graph-based iterative algorithm motivated by the previous success on message passing techniques and belief propagation algorithms. We compared the performance of ITRM with EigenTrust [21] and the Bayesian reputation management framework [10] (which is also proposed as the reputation management system of the well-known CONFIDANT protocol [9]) in a realistic DTN environment in Section 3.5 and showed the effectiveness of our proposed scheme.

Several works in the literature have focused on securing DTNs. In [24], the challenges of providing secure communication in DTNs is discussed and the use of Identity-Based Cryptography (IBC) [25] is suggested. In [26], source authentication and anonymous communication as well as message confidentiality are provided using IBC. In [27], the use of packet replication is proposed to improve message delivery rate instead of using cryptographic techniques. We note that the existing techniques to secure DTNs are aimed to provide data confidentiality and authentication only. On the other hand, our proposed trust-based scheme provides malicious node detection and high data availability with low packet latency in the presence of Byzantine attacks.

2 Iterative Trust and Reputation Management Mechanism (ITRM)

In this section, we describe ITRM and its security evaluation in a broader context (i.e., in a general setting). Then, we will modify and utilize it for DTNs in Section 3. Further, we will evaluate ITRM and compare its performance with some well-known reputation management techniques (e.g., Bayesian framework and EigenTrust) in a realistic DTN setting in Section 3.5. As in every trust and reputation management mechanism, we have two main goals: 1) computing the service quality (reputation) of the peers who provide a service (henceforth referred to as Service Providers or SPs) by using the feedbacks from the peers who used the service (referred to as the raters), and 2) determining the trustworthiness of the raters by



Iteration	TR_1	TR_2	TR_3
0	4.8	3	2.75
1	4.8	3.5	3.33
2	4.8	4.33	4.5
3	4.75	5	4.5

Iteration	C1	C2	C3	C4	C5	C6	C7
0	1.1	.72	.10	1.52	1.1	1.87	1.87
1	.85	.43	.35	1.23	.85	2.42	-
2	.43	.35	.77	.65	.43	-	-
3	.12	.38	-	.63	.12	-	-

(c)

Fig. 1. Illustrative example of ITRM.

analyzing their feedback about SPs. We consider the following major attacks that are common for any trust and reputation management mechanisms: 1) Bad mouthing, in which malicious raters collude and attack the SPs with the highest reputation by giving low ratings in order to undermine them, and 2) Ballot stuffing, in which malicious raters collude to increase the reputation values of peers with low reputations. Further, we evaluated ITRM against some sophisticated attacks (which utilizes bad mouthing or ballot stuffing with a strategy) such as RepTrap [28] or the one in which malicious raters provide both reliable and malicious ratings to mislead the algorithm.

Our proposed iterative algorithm is inspired by the earlier work on the improved iterative decoding algorithm of LDPC codes in the presence of stopping sets [4], [29]. In iterative decoding of LDPC, every check vertex (in the graph representation of the code) has some opinion of what the value of each bit vertex should be. The iterative decoding algorithm would then analyze the collection of these opinions to decide, at each iteration, what value to assign for the bit vertex under examination. Once the values of the bit vertices are estimated, in the next iteration, those values are used to determine the satisfaction probability of the check vertices values. The novelty of ITRM stems from the observation that a similar approach can be adapted to determine SPs' reputation values as well as the trustworthiness of the raters.

We let TR_j be the global reputation of the jth SP. Further, TR_{ij} represents the rating that the peer i reports about the SP j, whenever a transaction is completed between the two peers. Moreover, R_i denotes the (report/rating) trustworthiness of the ith peer as a rater.[1] The first step in developing ITRM is to interpret the collection of the raters and the SPs together with their associated relations as a bipartite graph, as in Fig. 1a. In this representation, each rater corresponds to a *check vertex* in the graph, shown as a square and each SP is represented by a *bit vertex* shown as a hexagon in the graph. If a rater i has a rating about the jth SP, we place an edge with value TR_{ij} from the ith check vertex to the jth bit vertex. As time passes, we use the age-factored values as the edge values instead. To each edge $\{ij\}$, a value $WR_{ij} = w_{ij}TR_{ij}$ is assigned, where WR_{ij} is the age-factored TR_{ij} value. The factor $w_{ij}(t)$ is used to incorporate the time-varying aspect of the reputation of the SPs (i.e., time-varying service quality). We use a known factor $w_{ij}(t) = \hat{\lambda}^{t-t_{ij}}$ where $\hat{\lambda}$ and t_{ij} are the fading parameter and the time when the last transaction between the rater i and the SP j occurred, respectively. If a new rating arrives from the ith rater about the jth SP, our scheme updates the new value of the edge $\{ij\}$ by averaging the new rating and the old value of the edge multiplied with the fading factor.

We consider slotted time throughout this discussion. At each time slot, ITRM will be executed using the input parameters R_i and WR_{ij} to obtain the reputation parameters (e.g., TR_j) and the list of malicious raters (referred to as the blacklist). Initially, the blacklist is set empty. Details of ITRM may be described by the following procedure at the Lth time slot. Let R_i and TR_{ij} be the parameter values prior to the present execution (the Lth execution) of ITRM algorithm. Let also TR_j^ν and TR_{ij}^ν be the values of the bit vertex and the $\{ij\}$th edge at the iteration ν of the ITRM algorithm. Prior to the start of the iteration ($\nu = 0$), we set $TR_{ij}^{\nu=0} = TR_{ij}$ and compute the initial value of each bit vertex (referred to as the initial guess $TR_j^{\nu=0}$) based on the weighted average of the age-factored edge values (WR_{ij}^ν) of all the edges incident to the bit vertex j. Equivalently, we compute

$$TR_j^\nu = \frac{\sum_{i\in A_j} R_i \times WR_{ij}^\nu}{\sum_{i\in A_j} R_i \times w_{ij}(t)}, \tag{1}$$

where A_j is the set of all check vertices connected to the bit vertex j. It is interesting to note that the initial guess values resemble the received information from the channel in the channel coding problem. Then, the first iteration starts (i.e., $\nu = 1$). We first compute the average inconsistency factor C_i^ν of each check vertex i using the values of the bit vertices (i.e., $TR_j^{\nu-1}$) for which it is connected to. That is, we compute

$$C_i^\nu = \left[1 \Big/ \sum_{j\in B} \hat{\lambda}^{t-t_{ij}}\right] \sum_{j\in B} d\big(TR_{ij}^{\nu-1}, TR_j^{\nu-1}\big), \tag{2}$$

where B is the set of bit vertices connected to the check vertex i and $d(\cdot,\cdot)$ is a distance metric used to measure the inconsistency. We use the $\mathcal{L}^1$ norm (absolute value) as the distance metric, and hence,

$$d\big(TR_{ij}^{\nu-1}, TR_j^{\nu-1}\big) = \big|TR_{ij}^{\nu-1} - TR_j^{\nu-1}\big| \hat{\lambda}^{t-t_{ij}}. \tag{3}$$

After computing the inconsistency factor for every check vertex, we list them is ascending order. Then, the check vertex i with the highest inconsistency is selected and placed in the blacklist if its inconsistency is greater than or equal to a definite threshold τ (whose choice will be discussed later). If there is no check vertex with inconsistency greater than or equal to τ, the algorithm stops its iterations. Once the check vertex i is blacklisted, we delete its rating TR_{ij}^ν for all the bit vertices j it is connected to.

1. All of these parameters (TR_j, TR_{ij} and R_i) may evolve with time. However, for simplicity, we omitted time dependencies from the notation.

Then, we update the values of all the bit vertices using (1). This completes the first iteration of ITRM. The iterative algorithm proceeds to other iterations exactly in the same way as the first iteration, updating the values of the bit vertices and blacklisting some other check vertices as a result. However, once a check vertex is placed in the blacklist, for the remaining iterations it is neither used for the evaluation of TR_js nor for the inconsistency measure of the check vertices. We stop the iterations when the inconsistencies of all the check vertices (excluding the ones already placed in the blacklist) fall below τ.

As an example, ITRM is illustrated in Fig. 1 for seven raters, three SPs, and $\tau = 0.7$. It is assumed that the rates are integer values from $\{1, \ldots, 5\}$ and the actual reputations, $T\hat{R}_j$, are equal to 5. For simplicity, we assumed w_i's to be equal to 1 and R_i's to be equal for all raters. Furthermore, we assumed that the peers 1, 2, 3, 4, and 5 are honest but 6 and 7 are malicious raters. The malicious raters (6 and 7) mount the bad-mouthing attack in this example. Fig. 1a shows the TR_{ij} values (illustrated by different line styles) prior to the execution of ITRM. The TR_j values and the individual inconsistencies of the raters after each iteration are also illustrated in Fig. 1c. We note that the algorithm stops at the third iteration when all the raters have inconsistencies less than τ. Fig. 1c indicates how ITRM gives better estimates of TR_j's compared to the weighted averaging method (which is correspond to the zero iteration). Fig. 1b illustrates the edges after the final iteration of ITRM. It is worth noting that the malicious raters 6 and 7 are blacklisted and their ratings are accordingly deleted. Moreover, rater 3, although honest, is also blacklisted at the third iteration. We note that this situation is possible when an honest but faulty rater's rating have a large deviation from the other honest raters.

2.1 Raters' Trustworthiness

We update the R_i values using the set of all past blacklists together in a Beta distribution. Initially, prior to the first time-slot, for each rater peer i, the R_i value is set to 0.5 ($\phi_i = 1$ and $\varphi_i = 1$). Then, if the rater peer i is blacklisted, R_i is decreased by setting

$$\varphi_i(t+1) = \bar{\lambda}\varphi_i(t) + (C_i + 1 - \tau)^{\delta}, \tag{4}$$

otherwise, R_i is increased by setting

$$\phi_i(t+1) = \bar{\lambda}\phi_i(t) + 1, \tag{5}$$

where $\bar{\lambda}$ is the fading parameter and δ denotes the penalty factor for the blacklisted raters. We note that updating R_i values via the Beta distribution has one major disadvantage. An existing malicious rater with low R_i could cancel its account and sign in with a new ID (*whitewashing*). This problem may be prevented by updating R_i's using the method proposed in [30].

2.2 Security Evaluation of ITRM

To prove that the general ITRM framework is a robust trust and reputation management mechanism, we briefly evaluate its security both analytically and via computer simulations. Then, in Section 3.5, we will evaluate the security of ITRM in a realistic DTN environment. In order to facilitate future references, frequently used notations are listed below:

- D Number of malicious raters
- H Number of honest raters
- N Number of service providers
- m Rating given by an honest rater
- n Rating given by a malicious rater
- X Total number of malicious ratings TR_{ij} per a victim SP
- d Total number of newly generated ratings, per time-slot, by an honest rater
- b Total number of newly generated ratings, per time-slot, by a malicious rater
- $\hat{b}$ Total number of newly generated attacking/malicious ratings, per time-slot, by a malicious rater
- Δ $\hat{b}/b$ (i.e., fraction of attacking ratings per time-slot)
- μ Total number of un-attacked SPs rated by an honest rater

2.2.1 Analytic Evaluation

We adopted the following models for various peers involved in the reputation system. We assumed that the quality of SPs remains unchanged during time slots. We provided the evaluation for the bad-mouthing attack only, as similar results hold for ballot stuffing and combinations of bad mouthing and ballot stuffing. We let $T\hat{R}_j$ be the actual reputation value of the jth SP. Ratings (i.e., TR_{ij}) generated by the nonmalicious raters are distributed uniformly among the SPs. We further assumed that m is a random variable with folded normal distribution (mean $T\hat{R}_j$ and variance 0.5); however, it takes only discrete values from 1 to 5. Furthermore, the values of R_i for all the raters are set to the highest value (i.e., $R_i = 1$) for simplicity (which reflects the worst case). Finally, we assumed that d is a random variable with Yule-Simon distribution, which resembles the power-law distribution used in modeling online systems, with the probability mass function $f_d(d;\rho) = \rho B(d, \rho + 1)$, where $B(\cdot,\cdot)$ is the Beta function. For modeling the adversary, we made the following assumptions. We assumed that the malicious raters initiate bad mouthing and collude while attacking the SPs. Further, the malicious raters attack the same set Γ of SPs at each time slot. In other words, Γ represents a set of size $\hat{b}$ in which each SP has an incoming edge from all malicious raters. The following discussions are developed for the time slot t.

τ-eliminate-optimal scheme. We declare a reputation scheme to be τ-eliminate-optimal if it can eliminate all the malicious raters whose inconsistency (measured from actual reputation values $T\hat{R}_j$ of SPs) exceeds the threshold τ. Hence, such a scheme would compute the reputations of the SPs by just using the honest raters. Naturally, we need to answer the following question: for a fixed τ, what are the conditions to have a τ-eliminate-optimal scheme? The conditions for ITRM to be a τ-eliminate-optimal scheme are given by the following lemma:

Lemma 1. *Let Θ_j and d_t be the number of unique raters for the jth SP and the total number of outgoing edges from an honest rater in t elapsed time slots, respectively. Let also Q be a*

random variable denoting the exponent of the fading parameter $\hat{\lambda}$ at the tth time slot. Then, ITRM would be a τ-eliminate-optimal scheme if the conditions

$$\sum_{r\in\Lambda}\Psi_r \geq (\hat{b}m + b\tau) \tag{6a}$$

and

$$\frac{\mu}{d_t} > 1 - \frac{\Theta\hat{\lambda}^Q\Delta}{D} \tag{6b}$$

are satisfied at the tth time slot, where

$$\Psi_r = \frac{mX + n\Theta_r\hat{\lambda}^Q}{X + \Theta_r\hat{\lambda}^Q} \text{ for } r \in \Lambda, \tag{7}$$

and Λ is the index set of the set Γ.

Proof. At each iteration, ITRM blacklists the rater i with the highest inconsistency C_i if $C_i \geq \tau$. Each malicious rater has $\hat{b}$ attacking ratings at each time slot. Moreover, the inconsistency of a malicious rater due to each of its attacking edge j is $\left(\frac{mX+n\Theta_j\hat{\lambda}^Q}{X+\Theta_j\hat{\lambda}^Q} - m\right)$, where $j \in \Gamma$. Therefore, the total inconsistency of a malicious rater (which is calculated considering both its attacking and nonattacking ratings) should be greater than or equal to τ to be blacklisted. This results the condition in (6a). Further, given $C_i \geq \tau$ for a malicious rater i, to have a τ-eliminate-optimal scheme, we require that the inconsistency of the malicious rater with the highest inconsistency exceeds the inconsistencies of all the reliable raters so that the blacklisted rater can be a malicious one in all iterations. To make sure ITRM blacklists all malicious raters, the inconsistency of a malicious rater must be greater than the inconsistency of a reliable rater at the 0th iteration with a high probability. The inconsistency of a malicious rater at the tth time slot is given by

$$\left(\left|\frac{mX + nc\lambda^Q}{X + c\lambda^Q} - m\right|\right)\Delta. \tag{8}$$

Similarly, the inconsistency of a reliable rater at the tth time slot is

$$\left(\left|\frac{mX + nc\lambda^Q}{X + c\lambda^Q} - n\right|\right)\frac{d_t - \mu}{d_t}. \tag{9}$$

Hence, to blacklist a malicious rater, we require the term in (8) be greater than that of (9) which leads to (6b). □

The design parameter τ should be selected based on the highest fraction of malicious raters to be tolerated. To determine the optimal value of τ, we start with Lemma 1. We use a waiting time t such that (6a) and (6b) are satisfied with high probability (given the highest fraction of malicious raters to be tolerated). Then, among all τ values that satisfy (6a) and (6b) with high probability, we select the highest τ value. The intention for selecting the highest τ value is to minimize the probability of blacklisting a reliable rater. In the following example, we designed the scheme to tolerate up to $W = 0.30$ (i.e., 30 percent malicious raters). For the given parameters $D + H = 200$, $N = 100$, $\Delta = 1$, $\rho = 1$, and $\hat{\lambda} = 0.9$, we obtained the optimal $\tau = 0.4$. As shown in Fig. 2, for W lower than 0.30, the waiting time becomes shorter to have a τ-eliminate-optimal scheme for $\tau = 0.4$. However, the scheme may also blacklist a few nonmalicious raters in addition to the malicious ones when W is actually less than 0.30. This is because the optimal value of τ is higher for a τ-eliminate-optimal scheme when W is actually less than 0.30.

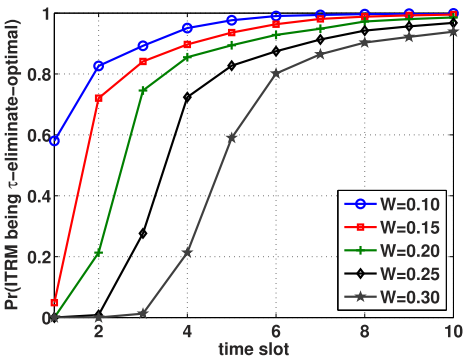


Fig. 2. Waiting time for τ-eliminate-optimal.

2.2.2 Simulations

We evaluated the performance of ITRM via computer simulations. We assumed that there were already 200 raters (all of which are honest and provide reliable ratings) and 50 SPs in the system. Moreover, a total of 50 time slots have passed since the launch of the system. Further, ratings generated during previous time slots were distributed among the SPs in proportion to their reputation values. After this initialization process, we introduced 50 more SPs as newcomers. Further, we assumed that a fraction of the existing raters changed behavior and became malicious after the initialization process. Hence, by providing reliable ratings during the initialization period (for 50 time slots) the malicious raters increased their trustworthiness values before they attack. Eventually, we had $D + H = 200$ raters and $N = 100$ SPs in total. We further assumed that d is a random variable with Yule-Simon distribution as discussed in the analysis. At each time slot, the newly generated ratings from honest raters are assigned to the SPs in proportion to the present estimate of their reputation values, TR_j. We obtained the performance of ITRM, for each time slot, as the mean absolute error (MAE) $|TR_j - \hat{TR}_j|$, averaged over all the SPs that are under attack (where, $\hat{TR}_j$ is the actual value of the reputation). We used the following parameters throughout our simulations: $b = 5$, $\rho = 1$, $\hat{\lambda} = \bar{\lambda} = 0.9$, the penalty factor $\delta = 10$, and $\tau = 0.4$ (the choice of τ is based on the analytical results discussed in Section 2.2.1).

We have evaluated the performance of ITRM in the presence of bad mouthing and ballot stuffing. Here, we provide an evaluation of the bad-mouthing attack only, as similar results hold for ballot stuffing. In all simulations, we considered the worst case scenario in which the victims are chosen among the newcomer SPs with an actual reputation value of $\hat{TR}_j = 5$ in order to have the most adverse effect. The malicious raters do not deviate very much from the actual $\hat{TR}_j = 5$ values to remain under cover as many time slots as possible (while still attacking). Hence, at each time

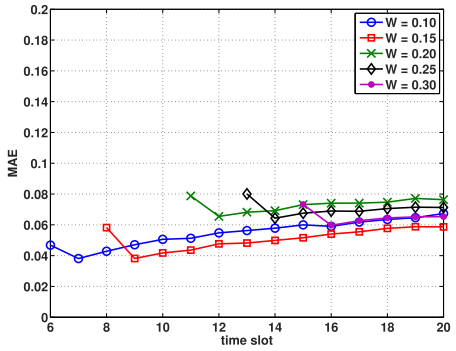


Fig. 3. MAE performance of ITRM for bad mouthing and for varying W.

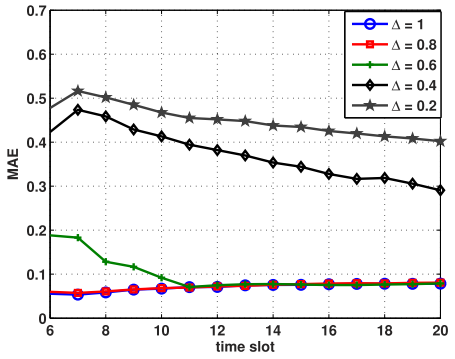


Fig. 4. MAE performance of ITRM for bad mouthing when $W = 0.10$ and for varying Δ.

slot, the malicious raters apply a low intensity attack by choosing the same set of SPs from Γ and rate them as $n = 4$. We had also tried higher deviations from the $T\hat{R}_j$ value and observed that the malicious raters were easily detected by ITRM in fewer time slots. Therefore, we identified the low-intensity attack scenario as the most adverse one against the reputation management mechanism. We note that this attack scenario also resembles the RepTrap attack in [28] which is proved to be a strong and destructive attack that can undermine the reputation system. Further, by assuming that the ratings of the reliable raters deviate from the actual reputation values, our attack scenario becomes even harder to detect when compared to the RepTrap. Fig. 3 illustrates the MAE performance of ITRM for this attack scenario after the newcomer SPs joined to the system and varying fractions of existing raters (W) changed behavior and became malicious. Thus, the plots in Fig. 3 are shown from the time slot the newcomers are introduced and existing raters changed behavior. We note that for this simulation we set $\Delta = \hat{b}/b = 1$. The lags in the plots of ITRM in Fig. 3 correspond to waiting times to include the newcomer SPs into the execution of ITRM, computed based on our analytical results presented in Fig. 2. We also observed that the average number of iterations for ITRM is around 5 and it decreases with time and with decreasing fraction of malicious raters.

We also evaluated the performance of ITRM when the malicious raters provide both reliable and malicious ratings to mislead the algorithm. In Fig. 4, we illustrate the performance of ITRM for this attack for $W = 0.10$ and different $\Delta = \hat{b}/b$ values. We observed that as the malicious raters attack with less number of edges (for low values of $\hat{b}$), it requires more time slots to undo their impact using ITRM. Further, when the $\hat{b}$ values becomes very small ($\hat{b} = 1, 2$), it is hard to detect the malicious peers. On the other hand, although the malicious raters stay under cover when they attack with very less number of edges, this type of an attack limits the malicious raters' ability to make a serious impact (they can only attack to a small number of SPs). It is worth noting that Fig. 4 only considers the MAE on the SPs that are under attacked. Thus, if the MAE is normalized over all SPs, it becomes clear that the impact of the malicious raters is reduced as they attack using smaller $\hat{b}$ values. We note that for small values of $\hat{b}$, other reputation management mechanisms also fail to detect the malicious raters. From these simulation results, we conclude that ITRM framework provides robust trust and reputation management in the presence of attacks.

3 Trust Management and Adversary Detection in DTNs

3.1 Adversary Models and Security Threats

As discussed in Section 1, we consider the challenging problem of countering Byzantine (insider) attacks (that give serious damage to the network in terms of data availability, latency, and throughput). We note that the security issues such as source authentication and data authentication have been considered for disconnected networks in [24], [26]. Hence, they are not explicitly considered in this paper. Instead, broadly we consider two types of attack: 1) attack on the network communication protocol, 2) attack on the security mechanism.

Packet drop and packet injection (attack on the network communication protocol). An insider adversary drops legitimate packets it has received. This behavior of the malicious nodes has a serious impact on the data availability and the total latency of the network. Moreover, a malicious node may also generate its own flow to deliver to another (malicious) node via the legitimate nodes. As a result, bogus flows compete with legitimate traffic for the scarce network resources.

Bad mouthing (ballot stuffing) on the trust management (attack on the security mechanism). As it will be discussed, a legitimate node needs feedbacks from a subset of nodes to determine its trust on a specific node. When a malicious node is an element of this subset, it gives incorrect feedback in order to undermine the trust management system. Bad-mouthing and ballot-stuffing attacks attempt to reduce the trust on a victim node and boost the trust value of a malicious ally, respectively. A successful attack may result in an incorrect edge value (rating) from a nonmalicious check vertex in the graph representation in Fig. 1a.

Random attack on trust management (attack on the security mechanism). A Byzantine node may adjust its packet drop rate (on the scale of zero-to-one) to stay under cover, making it harder to detect.

Bad mouthing (ballot stuffing) on the detection scheme (attack on the security mechanism). As it will be discussed, every legitimate node, in order to detect the nature of every network node, creates its own trust entries in a table (referred to as the node's rating table) for a subset of network nodes for which the node has collected sufficient feedbacks. Further, each node also collects rating tables from other nodes. When the Byzantine nodes transfer their

tables to a legitimate node, they may victimize the legitimate nodes (in the case of bad mouthing) or help their malicious allies (in the case of ballot stuffing) in their rating table entries. This effectively reduces the detection performance of the system. Furthermore, malicious nodes can provide both reliable and malicious ratings to mislead the algorithm as discussed in Section 2.2.2. A successful attack adds a malicious check vertex providing malicious edges (ratings) in the graph representation in Fig. 1a.

During the evaluation of the proposed scheme, we assumed that malicious nodes may mount attacks on both the network communication protocol and the underlying security mechanism (trust and reputation management mechanism, ITRM) simultaneously. In the attack on the network communication protocol, we assumed that malicious nodes both drop the legitimate packets they have received from reliable nodes and generate their own flows to deliver to other (malicious) nodes via the legitimate nodes in order to degrade the network performance (i.e., data availability and packet delivery ratio) directly. In the attack on the security mechanism, we assumed that malicious nodes simultaneously execute "bad mouthing (ballot stuffing) on the trust management," "random attack on trust management," and "bad mouthing (ballot stuffing) on the detection scheme" (which are described above) to cheat the underlying trust and reputation management scheme (i.e., ITRM) and degrade the network performance indirectly. We study the impact of these attacks and evaluate our proposed scheme in the presence of these attacks (on the network communication protocol and the security mechanism) in Section 3.5. First, we study the impact of the attacks to cheat the underlying trust and reputation management mechanism alone and obtain the time required to detect all the malicious nodes in the network. Next, we study the impact of the "packet drop and packet injection attack" to the network performance (in terms of data availability and packet delivery ratio) while the malicious nodes also mount attacks on the underlying reputation mechanism.

As a result of our studies, we concluded that ITRM provides a very efficient trust management and malicious node detection mechanism for DTNs under the threat model discussed above. The most significant advantage of ITRM under the above threat model, in addition to resiliency to a high fraction of malicious nodes, is to let each network node accurately compute the reputation values of the other network nodes in a short time. Computing the reputation values in a short time is a very crucial issue in DTNs because of their unique characteristics (such as the intermittent contacts between the nodes). As a result of this advantage, each legitimate node detects and isolates the malicious nodes from the network to minimize their impact to the network performance (as will be illustrated in Section 3.5).

We note that since we did not assume preexisting trust relationships among the nodes, we did not study some particular attacks such as RepTrap [28] (which is studied in Section 2.2.2 to evaluate the performance of ITRM) particularly for DTNs.

3.2 Network/Communication Model and Technical Background in Context

Before giving a high-level description of our scheme, we will introduce the network/communication model and the main tools that we use for the system to operate.

Mobility model. We use both Random Waypoint (RWP) and Levy-walk (LW) mobility models for our study which are widely used for simulating DTNs. RWP model produces exponentially decaying intercontact time distributions for the network nodes making the mobility analysis tractable. On the other hand, LW is shown to produce power-law distributions that has been studied extensively for animal patterns and recently has been shown to be a promising model for human mobility [31]. In the RWP mobility model [32], each node is assigned an initial location in the field and travels at a constant speed to a randomly chosen destination. The speed is randomly chosen from $[v_{min}, v_{max}]$ independently of the initial location and destination. After reaching the destination, the node may pause for a random amount of time before the new destination and speed are chosen randomly for the next movement. In LW mobility model [31], [33], [34], on the other hand, each movement length and pause time distributions closely match truncated power-law distributions. Further, angles of movement are pulled from a uniform distribution. Our implementation of the LW mobility model is based on the model in [31]. A step is represented by four variables, movement length (ℓ), direction (θ), movement time (Υt_f), and pause time (Υt_p). The model selects movement lengths and pause times randomly from their Levy distributions $p(\ell)$ and $\psi(\Upsilon t_p)$ with coefficients α and β, respectively. Finally, regardless of the mobility model used, we assume a finite rate of packet transfer which forces the number of packets transmitted per contact to be directly proportional to the contact time.

Packet format. We require that each packet contains its two hop history in its header. In other words, when node B receives a packet from node A, it learns from which node A received that packet. This mechanism is useful for the feedback mechanism as discussed in Section 3.4.

Routing and packet exchange protocol. We assume that messages at the source are packetized. Further, the source node never transmits multiple copies of the same packet. Hence, at any given time, there is at most a single copy of each packet in the network. We assume only single-copy routing since reliable single-copy routing with packetization is achieved by encoding the data packets using rateless codes [35], [36] at the source node. The use of rateless coding improves reliability and latency in DTNs even when there is no adversary [37]. Furthermore, exchange of packets between two nodes follows a back-pressure policy. To illustrate this, assume nodes A and B have x and y packets belonging to the same flow f, respectively (where $x > y$). Then, if the contact duration permits, node A transfers $(x - y)/2$ packets to node B belonging to flow f. As a result of the mobility model, each node has the same probability to meet with the destination of a specific flow. Hence, by using the back-pressure policy, we equally share the resources (e.g., contact time) among the flows.

The packet exchange protocol also enforces fairness among multiple nodes that forwarded the same flow to a node. To clarify, let us assume that node A has some packets from a flow f (which were forwarded to it by χ different nodes) and based on the back-pressure policy, it needs to transfer some of them to node B. In this situation, node A must fairly select the packets based on their previous hops (which is available via the packet format discussed before). In other words, each packet that is received from a different node has the same probability to be selected for transfer. This mechanism is useful for the feedback mechanism as discussed later. Finally, when a node forwards a packet, it deletes it from its buffer.

Bloom filter. A Bloom filter is a simple space-efficient randomized data structure for representing a set in order to support membership queries [38]. A Bloom filter for representing a set U of G elements is described by an array of κ bits, initially all set to 0. It employs γ independent hash functions $\mathbb{H}_1, \ldots, \mathbb{H}_\gamma$ with range $\{1, \ldots, \kappa\}$. For every element $x \in U$, the bits $\mathbb{H}_1(x), \ldots, \mathbb{H}_\gamma(x)$ in the array are set to 1. A location can be set to 1 multiple times, but only the first change has an effect. To check if y belongs to U, we check whether all $\mathbb{H}_1(y), \ldots, \mathbb{H}_\gamma(y)$ are set to 1. If not, y definitely does not belong to U. Otherwise, we assume $y \in U$ although this may be wrong with some probability. Hence, a Bloom filter may yield a *false positive* where it suggests that y is in U even though it is not. The network designer can arbitrarily decrease this probability to the expense of increasing communication overhead. Further, the false positive probability can be significantly reduced by using recently proposed techniques such as [39].

3.3 Iterative Detection for DTNs

In this section, we will describe how ITRM is adapted in DTNs as an iterative malicious node detection mechanism. We will pick an arbitrary node in the network and present the algorithm from its point of view throughout the rest of this paper. We denote this node as a *judge* for clarification of our presentation. Further, the counterpart to the quality of a SP in the discussion of ITRM is the reliability of the node in DTN in faithfully following the network (routing) protocols to deliver the packets.

Since direct monitoring is not an option in DTNs (as explained in Section 1.1), a judge node creates its own rating about another network node by collecting feedbacks about the node and aggregating them. Each judge node has a table (referred to as a *Rating Table*) whose entries (which are obtained using the feedback mechanism described in Section 3.4) are used for storing the ratings of the network nodes. In DTNs, due to intermittent contacts, a judge node has to wait for a very long time to issue its own ratings for all the nodes in the network. However, it is desirable for a judge node to have a fresh estimate of the reputation of all the nodes in the network in a timely manner, mitigating the effects of malicious nodes immediately. To achieve this goal, we propose an iterative detection mechanism which operates by using the rating tables formed by other nodes (acting as judges themselves). The rating table of a judge node can be represented by a bipartite graph consisting one check vertex (the judge node) and some bit vertices (i.e., a subset of all the nodes in the network for which the judge

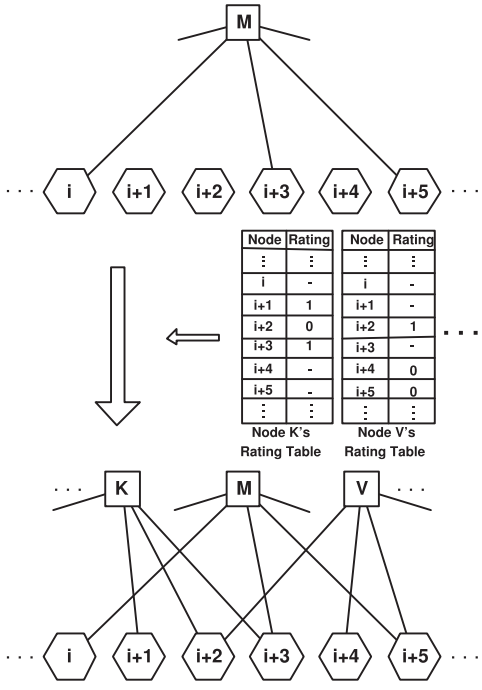


Fig. 5. Collecting and combining the rating tables at the judge node M.

node has received sufficient number of feedbacks to form a rating with high confidence). Besides, by collecting sufficient number of rating tables from other nodes, a judge node can generate a bipartite graph as in Section 2, which includes all the network nodes as bit vertices. We illustrate this process at judge node M in Fig. 5 in which node M collects rating tables from other judge nodes (including K and V) and generates a bipartite graph including all network nodes as bit vertices. Assuming N nodes in the network, a judge node may create a bipartite graph with N bit vertices by collecting rating tables from $k-1$ nodes each with at least s nonempty entries. Hence, the resulting graph would have k check vertices (the kth check vertex belongs the judge node). The parameters s and k are to be determined for high probability of detection while minimizing detection latency. Clearly, higher s and k reduces the detection error but increases the delay. We will discuss this issue in Section 3.5. Hence, when two nodes establish a contact in a DTN, they exchange their rating tables. Once a judge node collects sufficient number of tables each with sufficient number of nonempty entries, it can then proceed with the iterative algorithm to specify the reputation values for all the nodes.

To adapt the ITRM scheme for DTNs, we will present (feedback) ratings as "0" or "1," which results in binary reputation values. In this special case, the iterative reputation scheme becomes a detection scheme. That is, a node with a reputation value of zero would be interpreted as a malicious node. Therefore, the proposed scheme detects and isolates the malicious nodes from the network to minimize their impact. We note that we used binary rating values for simplicity of the setup. Alternatively, one may consider a setup where ratings are nonbinary. In this scenario, when two nodes establish a contact, they may exchange packets with some probability associated with their reputation values (i.e., they may exchange packets

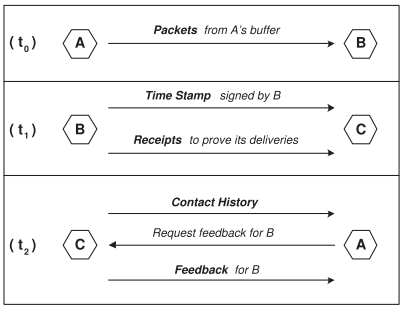


Fig. 6. Indirect type I feedback between nodes A (judge), B (suspect), and C (witness).

proportional to their reputation values). Moreover, we did not incorporate R_i values for simplicity of simulations, and hence, we set all R_i values to one for the application of ITRM in DTNs. In other words, we assume that the judge node does not have any previous knowledge about the witness nodes and it trusts each witness node equally.

3.4 Trust Management Scheme for DTNs

In the proposed scheme, the authentication mechanism for the packets generated by a specific source is provided by a Bloom filter [38] and ID-based signature (IBS) [25]. Whenever a source node sends some packets belonging to the flow that is initiated by itself, it creates a Bloom filter output from those packets, signs it using IBS, and sends it to its contacts. The Bloom filter output provides an authentication mechanism for the packets generated by a specific source. It is worth noting that whenever an intermediate node forwards packets belonging to a specific flow to its contact, it also forwards the signed Bloom filter output belonging to those packets for the packet level authentication at each intermediate node. We do not give further details of the authentication mechanism as source and data authentication for DTNs have been considered before [24], [26] and they are out of the scope of this paper.

Our proposed feedback mechanism to determine the entries in the rating table is based on a 3-hop loop (referred to as *Indirect type I* feedback). We will describe this scheme by using a toy example between three nodes A, B, and C as follows: let us denote the node that is evaluating as the *judge* (node A), the node that is being evaluated as the *suspect* (node B), and the node that was the direct contact of the suspect as the *witness* (node C). The basic working principle of the mechanism is that after the judge node has a transaction (in the form of passing some packets) with a suspect, the judge node waits to make contacts and receive feedback about the suspect from every node (i.e., witnesses) that has been in direct contact with the suspect. It is worth noting that this feedback mechanism is only used for constructing the entries in the judge node's rating table for a few network nodes. In overall, rating tables are collected from the contacts of the judge node and ITRM is applied to find the reputations of all network nodes (as described in Section 3.3).

Let assume that node A meets B, B meets C, and C meets A at times t_0, t_1, and t_2, respectively, where $t_0 < t_1 < t_2$. Indirect type I feedback between nodes A, B, and C is illustrated in Fig. 6. At time t_0, A and B execute mutual packet exchange as described in Section 3.2. When B and C meet at t_1, they first exchange signed time stamps. Hence, when C establishes a contact with A, it can prove that it indeed met B. Then, B sends the packets in its buffer executing the fairness protocol discussed in Section 3.2. Moreover, (suspect) node B transfers the receipts it received thus far to the (witness) C. Those receipts include the proofs of node B's deliveries (including deliveries of the packets belonging to node A) thus far and are signed by the nodes to which its packets were delivered. We note that the receipts expire in time and deleted from the buffers of the witnesses. Hence, they are not accumulated in the buffers of the nodes. The lifetime of the receipts are determined based on the detection performance of the scheme (required time for the scheme to have a high malicious node detection accuracy) as will be described in Section 3.5. At the end of the contact, node C also gives a signed receipt to node B including the IDs of the packets it received from B during the contact. Finally, when the judge node A and the witness C meet, they initially exchange their contact histories. Hence, A learns that C has met B and requests the feedback. The feedback consists of two parts: 1) those receipts of B that are useful for A's evaluation (i.e., receipts which include the delivery proofs of the packets belonging to node A), and 2) if node C received node A's packets from node B, it sends the hashes of those packets to A for the latter's evaluation. We note that C can easily find out A's packets by just examining the headers as explained in Section 3.2. From B's receipts, node A can determine if B followed the packet delivery procedure (which is described in Section 3.2) properly while delivering the packets forwarded by node A at time t_0 (B's receipts will reveal the packet deliveries of B after time t_0). Further, from the hashes of its own packets (if there is any received by node C), node A can determine if node B had modified any of the packets before delivery.

If both parts of the feedback are verified by node A (if node B followed the packet delivery procedure for A's packets and delivered the packets properly), then the judge A makes a "positive evaluation" as 1. Otherwise, if either part of the feedback is not verified, the evaluation will be "negative" as 0. We note that if node C did not receive any packets belonging to node A, then node A's evaluation will be only based on the receipts of B which are provided by node C at time t_2 (i.e., node A will evaluate node B based on the receipts it received from node C, which is the first part of the feedback explained before). We note that the feedbacks from the witnesses are not trustable. Because of the bad mouthing (ballot stuffing) and random attacks (discussed in Section 3.1), a judge node waits for a definite number of feedbacks to give its verdict about a suspect node with a high confidence. We will discuss this waiting time, the number of required feedbacks, and their interplay for different adversarial models in Section 3.5. Hence, each judge node uses the Beta distribution to aggregate multiple evaluations it has made about a suspect using the associated feedbacks to form its rating (verdict) for a suspect node. That is, if the aggregation of multiple feedbacks for a suspect node is bigger that 0.5, the suspect node is rated as "1" in the judge node's rating table (i.e., the node's verdict is

"1"). Otherwise, if the aggregation value is smaller than or equal to 0.5, the suspect node is rated as "0."[2]

In the high-level description of ITRM, it was implicitly assumed that the judge has a priori knowledge about the packet drop rate of the Byzantine node. This is unrealistic as the nodes may apply random attacks as in Section 3.1. To remove this assumption, we propose detection at different levels. We observed that the sufficient number of feedbacks that is required to give a verdict with high confidence depends on the packet drop rate of the Byzantine nodes. In other words, for a node with a higher drop rate, we would require fewer feedbacks than a node with a lower drop rate. Assume that we desire to perform detection at level $p_1 = 0.8$. This implies that after applying ITRM, each judge node would identify and isolate all the Byzantine nodes whose packet drop rates are p_1 or higher. Further, assume that the detection at level p_1 requires at least $\hat{M}_1$ feedbacks about a suspect node. The number of feedbacks depends on the confidence we seek at the accuracy of a verdict (before detection). The level of confidence is determined by the detection strategy. For instance, for ITRM, a confidence value in the order of 0.95 (out of 1) would be sufficient. Clearly, the number of feedbacks also depends on the detection level. The lower the detection level, the higher is the number of required feedbacks to maintain the same detection confidence. Hence, every judge stores together with its verdict the lowest level of detection at which the verdict can be used. Obviously, an entry verdict with lower detection level (e.g., $p = 0.6$) is also good for use in a high detection level (e.g., $p = 0.8$), but the inverse is not true. An entry is left empty if the judge does not have the sufficient number of feedbacks to give any verdict even at the highest detection level. We note that there is no predetermined detection level for the proposed scheme. The judge node applies the ITRM for the lowest possible detection level (to minimize the impacts of malicious nodes) depending on the entries (number of feedbacks used to construct each entry verdict) in both its own rating table and the rating tables it collected from other nodes. The judge checks the detection level of each table entry (from both its own table and the collected tables) and performs the ITRM at the detection level of the entry verdict which is the largest. To clarify this, assume a judge node M collected rating tables from other nodes K and V as in Fig. 5. For this toy example, we assume that the judge node M performs the ITRM by using only three rating tables (its own rating table and the ones collected from nodes K and V). We further assume that the rating table entries with the largest detection levels has a detection level of m, k, and v for nodes M, K, and V's rating tables, respectively. Then, the judge node M performs the ITRM at the detection level of $max(m, k, v)$. As a result of this mechanism, the malicious nodes may try to survive from the detection mechanism by setting their packet drop rates to lower values. However, the proposed detection mechanism eventually detects all the malicious nodes (even the ones with lower packet drop rates) when the judge node waits longer times to apply the ITRM at a lower detection level. Further, as the drop rate of the malicious nodes gets lower, the negative impact of the malicious nodes gets less significant in terms of data availability and packet delivery ratio.

3.5 Security Evaluation

In this section, we give an analysis of the metrics of interest and illustrate our simulation results. Further, we compare the performance of ITRM with the well-known reputation management schemes (Bayesian framework [10] and EigenTrust [21]) in a realistic DTN environment. Finally, we show the performance of the proposed scheme for the malicious node detection, availability, and packet delivery ratio via simulations (conducted using Matlab). We assumed the mobility models (RWP and LW) of Section 3.2 with N nodes in the network. It is shown that the intercontact time distributions of the LW can be modeled by a truncated Pareto distribution [34]. On the other hand, as we mentioned in Section 3.2, the fact that the intercontact times of the RWP mobility model can be modeled as a Poisson process [40] makes the mobility analysis tractable. Therefore, for our analytical conclusions (in Lemmas 2 and 3), we assumed the RWP mobility model.[3] However, for the simulations, we used both RWP and LW mobility models to evaluate the performance of the proposed scheme under different mobility models.

In all simulations, we fixed the simulation area to 4.5 km by 4.5 km (with reflecting boundaries) which includes $N = 100$ nodes each with a transmission range of 250 m (which is the typical value for IEEE 802.11b). For the RWP model, we used $[v_{min}, v_{max}] = [10, 30]$ m/s and ignored the pause time for the nodes. For the LW model, we set the speed of every node to 10 m/s. Further, we set the scale factors of movement lengths and pause times to 10 and 1, respectively. We used the Levy distribution coefficients of $\alpha = 1$ and $\beta = 1$. Finally, we set the maximum movement length and pause time to 4 km and 2 hours, respectively.

Confidence on a verdict. We let λ_i be the intercontact time between two particular nodes. We analytically illustrated the waiting time of a judge node to collect sufficient number of feedbacks about a suspect (to give its verdict with high confidence) and evaluated the effect of random attack on the required number of feedbacks in the following. Let the random variables x, y, and z represent the number of feedbacks received at a specific judge node A (about a suspect node B), total number of contacts that the suspect node B established after meeting A, and the number of distinct contacts of B after meeting A, respectively. The following lemma characterizes the time needed to receive M distinct feedbacks about a particular suspect node B at a particular judge node A for the RWP mobility model.

Lemma 2. *Let t_0 be the time that a transaction occurred between a particular judge-suspect pair. Further, let N_T be the number of feedbacks received by the judge for that particular suspect node since $t = t_0$. Then, the probability that the judge node has at least M feedbacks about the suspect node from M distinct witnesses at time $T + t_0$ is given by*

$$Pr(N_T \geq M) = \int_M^{\infty} \int_{-\infty}^{+\infty} f(x|z,T) f(z,T) dz dx. \quad (10)$$

2. ITRM then takes the rating tables, whose entries are associated verdicts, as inputs to process and determines the final faith of a node. Hence, the verdicts will be further examined by ITRM.

3. Similar results can be obtained for the LW mobility model using a truncated Pareto distribution for the intercontact times.

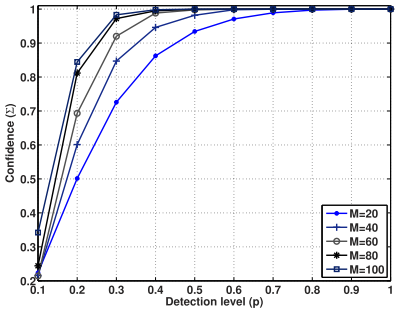


Fig. 7. Confidence of a judge node on its verdict versus the detection level for $W = 0.10$.

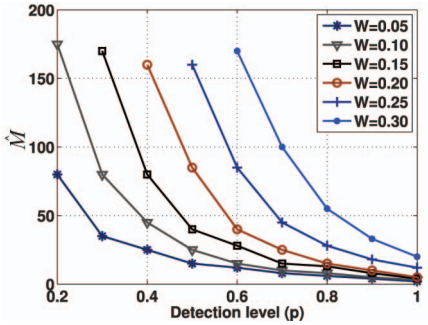


Fig. 8. $\hat{M}$ versus the detection level when $\Sigma = 0.95$ for different values of W.

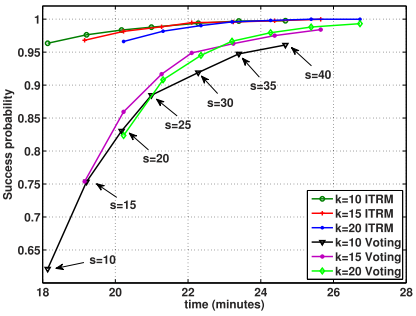


Fig. 9. Probability of detection success for fixed k and varying s values with the RWP mobility model.

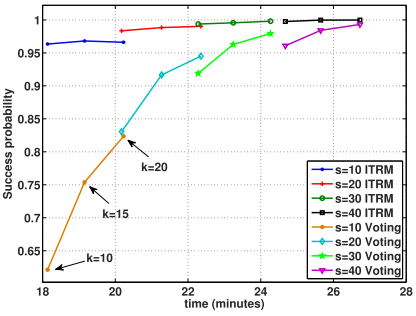


Fig. 10. Probability of detection success for fixed s and varying k values with the RWP mobility model.

Here, the distribution $f(x|z,t)$ is Poisson with rate $\lambda_i z/2$ and

$$f(z,t) = \int_{-\infty}^{+\infty} f(z|y,t) f(y,t) dy, \tag{11}$$

where $f(y,t)$ and $f(z|y,t)$ are both Poisson distributions with rates $(N-2)\lambda_i$ and $(N-2)\lambda_i - \lambda_i y/2$, respectively.

Proof. The probability that a particular judge node receives at least M feedbacks (from distinct witnesses) about a particular suspect node between time t_0 and $t_0 + T$ is given by

$$Pr(N_T \geq M) = \int_{M}^{\infty} f(x,T) dx, \tag{12}$$

where $f(x,t) = \int_{-\infty}^{+\infty} f(x|z,t) f(z,t) dz$. As a result of the RWP mobility model, it can be shown that $f(x|z,t)$ is Poisson with rate $\lambda_i z/2$ where z represents the number of distinct contacts of the suspect between time t_0 and $t_0 + T$ and x is the number of feedbacks received by the judge node (about the suspect) from a subset of those z contacts. Further, since there are N nodes in the network, it can be shown that the number of contacts established by any node has a Poisson distribution with rate $(N-1)\lambda_i$ (excluding itself). Therefore, the number of contacts the suspect established after the transaction with the judge, y, has a Poisson distribution with rate $(N-2)\lambda_i$ (excluding the judge node and the suspect node itself), and given y, the number of distinct contacts of the suspect z has a Poisson distribution with rate $(N-2)\lambda_i - \lambda_i y/2$. □

We studied the effect of random attack on the required number of feedbacks for a network with $N = 100$.[4] We denote the fraction of the Byzantine nodes in the network as W. As we discussed in Section 3.4, a judge node waits for a definite number of feedbacks to give its verdict about a suspect node with a high confidence. Fig. 7 illustrates the variation of a (judge) node's confidence Σ on its verdict for a suspect versus different levels of detection p. This is given for different number of feedbacks (M) when $W = 0.10$. As expected, a node has more confidence at higher detection levels and for high M values. Due to the bad mouthing, ballot stuffing and random attacks, a judge node must wait for a definite number of feedbacks to give its verdict about a suspect node with a high confidence. Let $\hat{M}$ be the minimum number of feedbacks required about a specific suspect node for an acceptable confidence level on a verdict. In Fig. 8, the variance of $\hat{M}$ for different detection levels (p) and different W values is illustrated for a judge node to have $\Sigma = 0.95$ confidence on its verdict (i.e., $\hat{M} = M$ for $\Sigma \simeq 0.95$). Using Fig. 8, we conclude that a judge node needs more feedbacks about a suspect when there are more malicious nodes mounting bad mouthing (or ballot stuffing) on the trust management.

Detection performance. We analytically obtained the waiting time of a judge node before executing ITRM and evaluated the effects of attacks on the detection scheme for a network of size N in which the intercontact time between two particular nodes is λ_i. Let $\hat{M}$ be the minimum number of feedbacks required about a specific suspect node for an acceptable confidence level on a verdict. Further, let $\hat{T}$ be the time required to receive $\hat{M}$ feedbacks for a specific suspect. The following lemma along with the simulation results illustrated in Figs. 9, 10, 11, and 12 (which will be presented next) provide a good insight for a judge node about the instant at which it should apply ITRM (the proof is similar to that of Lemma 2).

4. The results illustrated (in Figs. 7 and 8) are independent of the mobility model used.

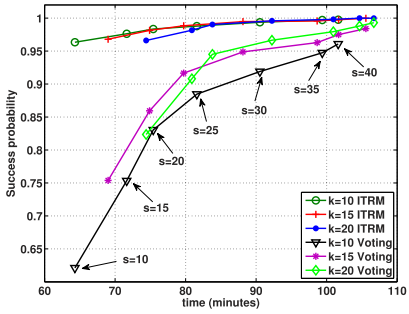


Fig. 11. Probability of detection success for fixed k and varying s values with the LW mobility model.

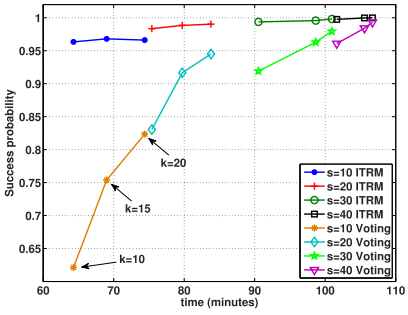


Fig. 12. Probability of detection success for fixed s and varying k values with the LW mobility model.

Lemma 3. *Let a particular judge node start collecting feedbacks and generating its rating table at time $t = t_0$. Further, let $\hat{N}_T$ be the number of entries in the rating table of the judge node. Then, the probability that the judge node has at least s entries at time $t_0 + T$ is given by*

$$Pr(\hat{N}_T \geq s) = \int_{s}^{+\infty} \int_{-\infty}^{+\infty} f(z|x, T - \hat{T}) f(x, T - \hat{T}) dx dz, \quad (13)$$

where $f(x,t)$ and $f(z|x,t)$ are Poisson distributions with the rates $(N-1)\lambda_i$ and $(N-1)\lambda_i - \lambda_i x/2$ for the RWP mobility model, respectively.

We evaluated the performance of ITRM for different (k, s) pairs (where k is the number of rating tables collected at the judge node and s is the number of nonempty entries in each table). Moreover, we compared ITRM with the well-known Voting Technique in which a judge node decides on the type of a suspect based on the majority of the votes for that node. For the Voting Technique, we used the Indirect type I feedback as described in Section 3.4 (since direct monitoring is not possible in DTNs, we believe that this feedback mechanism is the only option for the nodes). However, in the Voting Technique, instead of utilizing the ITRM, a judge node decides on the type of a suspect node based on the majority of feedbacks it received (i.e., a suspect node is identified as a malicious node if it received more negative feedbacks than the positive ones).

We defined the *success* of a scheme as its capability of detecting all malicious nodes in the network (without tagging any reliable node as malicious by mistake). We illustrated the probability of success S of ITRM and the Voting Technique for different (k, s) pairs versus the required time. We used both RWP and LW mobility models (with the parameters described previously) in our simulations. In both mobility models, whenever two nodes establish a contact, a transaction occurs between them in the form of the packet exchange. Further, it is assumed that the judge and malicious nodes start generating their rating tables and mounting their attacks at time $t = 0$, respectively.

We provide the evaluation only for the bad mouthing on the detection scheme and bad mouthing on the trust management only, as similar results hold for ballot stuffing and combinations of bad mouthing and ballot stuffing. In particular, malicious nodes provide incorrect feedbacks to the judge nodes about their reliable contacts in order to cause the judge nodes to misjudge the types of reliable nodes (in their verdicts). As a result of the malicious feedback, a judge node may make a "negative evaluation" (as described in Section 3.4) on a reliable node. Second, the malicious nodes collaboratively victimize the reliable nodes (i.e., attack the same set of reliable nodes) in their own rating tables by rating them as "0" and forward these rating tables whenever they contact with reliable nodes to mislead the detection mechanism.

In Fig. 9, we illustrated S versus time for fixed values of k and varying s for the RWP mobility model. In Fig. 10, the s values are fixed and the parameter k is varied with increments of 5 for the RWP model. Similarly in Figs. 11 and 12, we illustrated S for ITRM and the Voting Technique with the LW mobility model. In all figures, time is measured starting from $t = 0$. Our results support the fact that RWP shows a more optimistic routing performance compared to LW since its high occurrences of long movements intensify the chance of meeting destinations [31]. Further, these results also give some indication of the false positive (tagging a reliable node as malicious) and false negative (labeling a malicious node as reliable) probabilities of the proposed scheme as well. As S increases, the probability that the scheme detects all malicious nodes gets higher along with the probability that the scheme identifies all reliable nodes as reliable. Similarly, as S decreases, the probability that the scheme labels a malicious node as reliable gets higher along with the probability that the scheme marks a reliable node as a malicious one. In other words, false positive and false negative probabilities are high when the probability of success is low as in Figs. 9, 10, 11, and 12. Furthermore, these results can also be used to determine the lifetimes of the receipts at the witness nodes. Knowing how long it takes to have a high success probability at a judge node for a given detection level, the witnesses can delete the receipts which have been stored for more than the sufficient time required for a high success probability from their buffers. Based on our simulation results, we concluded that ITRM significantly outperforms the Voting Technique by providing higher success rates in shorter time (regardless of the mobility model) which is a very crucial issue in DTNs. We obtained these results for the fraction of malicious nodes W is 0.10 and for a detection level of $p = 0.8$. However, we note that the required (k, s) pairs to obtain a high success probability do not change with the detection level, which only has an effect on $\hat{M}$. It is worth noting that even though the time required to get the high success probability increases with increasing W,

the performance gap between ITRM and the Voting Technique remains similar for different values of W.

In the rest of this section, we will present our simulation results for different network parameters and show the performance of the proposed scheme for mean absolute error in the computed reputation values, data availability, and packet delivery ratio. We note that we did not compare the proposed scheme with existing DTN security schemes such as [26] since none of the existing schemes is aimed to provide data availability and malicious node detection as in our work. Further, it is worth noting that there is no existing trust and reputation management mechanism for DTNs. In spite of this, we compared the proposed scheme with the Bayesian reputation management framework in [10] (which is also proposed as the reputation management system of the well-known CONFIDANT protocol [9]) and the EigenTrust algorithm [21] in a DTN environment. For the Bayesian framework [10], we used the parameters from the original work [10] (deviation threshold $d = 0.5$ and trustworthiness threshold $t = 0.75$). Further, we set the fading parameter to 0.9 (for details refer to [10]). It is worth noting that neither the original Bayesian reputation framework in [10] nor EigenTrust [21] is directly applicable to DTNs since both protocols rely on direct measurements (e.g., watchdog mechanism) which is not practical for DTNs as discussed in Section 1.1. Therefore, we implemented [10] and [21] by letting the judge nodes collect indirect measurements (feedbacks) from the witnesses using Indirect type I feedback as described in Section 3.4. Since direct monitoring is not possible in DTNs, we believe that this feedback mechanism is the only option for the nodes. Thus, we assumed that, as in our scheme, each judge node collects feedbacks and forms its rating table. Further, each judge node exchanges its rating table with the other nodes upon a contact and then executes the reputation management protocol in [10] or EigenTrust [21]. We note that in principle, ITRM performs better than the Bayesian reputation management framework in [10] since Bayesian approaches of [10] and [41] assume that the reputation values of the nodes are independent. Hence, in these schemes, each reputation value is computed independent of the other nodes' reputation values using the ratings given to each node. However, this assumption is not valid because the ratings provided by the nodes induce a probability distribution on the reputation values of the nodes. However, this assumption is not valid because the ratings provided by the nodes induce a probability distribution on the reputation values of the nodes. These distributions are correlated because they are induced by the overlapping set of (rater) nodes. The strength of ITRM stems from the fact that it tries to capture this correlation in analyzing the ratings and computing the reputations. On the other hand, as we discussed in Section 1.1, the EigenTrust algorithm is constrained by the fact that trustworthiness of a peer (on its feedback) is equivalent to its reputation value. However, trusting a peer's feedback and trusting a peer's service quality are two different concepts since a malicious peer can attack the network protocol or the reputation management system independently. Therefore, in principle, ITRM also performs better than the EigenTrust algorithm. Indeed, our simulation results (presented next) also support these arguments.

We used the simulation settings described before with the LW mobility model. We assumed that a definite amount of time (4 hours) has elapsed since the launch of the system as the initialization period, during which new messages are generated by a Poisson distribution at rate $\lambda_m = 1/3{,}000$ at the source nodes and transmitted to their respective destinations. Further, during this initialization period, rating tables were being created at the judge nodes. Then, at time $t = 0$ (after the initialization period),[5] we assumed legitimate nodes simultaneously start new flows to their destinations (while the previous flows may still exist) and attackers start mounting their attacks (both on the network communication protocol and the security system). Therefore, at time $t = 0$, we assumed each legitimate source node has 1,000 information packets which are encoded via a rateless code for single-copy routing transmission. Hence, the number of encoded packets required by each destination to recover a message is roughly 1,000.[6] We assumed packets with 128 bytes payloads and a data rate of 250 kbps for each link. We note that we used the same routing and packet exchange protocol for ITRM, Bayesian framework and EigenTrust algorithm (which is described in Section 3.2). We evaluated the data availability and packet delivery ratio for these new flows since time $t = 0$. Moreover, we let each judge node execute ITRM, Bayesian framework, or EigenTrust algorithm starting from time $t = 0$, and hence, we also evaluated the MAE since time $t = 0$. Thus, for all simulations, the plots are shown from time $t = 0$. The percentage of the Byzantine nodes in the network is denoted as W. For ITRM, the Bayesian framework in [10], and EigenTrust [21], we assumed that each judge node randomly picks 10 entries from each rating table it received in order to prevent the malicious users from flooding the mechanism with incorrect entries. We ran each simulation 100 times to get an average. We executed the experiment with different parameters in the LW mobility model (e.g., different Levy distribution coefficients, node speeds, etc.) and obtained similar trends. We further simulated the proposed scheme with the RWP mobility model with $[v_{min}, v_{max}] = [10, 30]\ \mathrm{m/s}$ and ignoring the pause times. The RWP model resulted in similar trends as the LW model, and hence, we do not report its results due to the space limit.

As before, we present the evaluation only for the bad mouthing on the detection scheme and bad mouthing on the trust management (as described in Section 3.1), as similar results hold for ballot stuffing and combinations of bad mouthing and ballot stuffing. Malicious nodes provide incorrect feedbacks to the judge nodes about their reliable contacts in order to cause the judge nodes to misjudge the types of reliable nodes (in their verdicts). Further, malicious nodes collaboratively victimize the reliable nodes in their rating tables by rating them as "0" and forward their rating tables whenever they contact with a reliable node to

5. Once the initialization period is elapsed, we set the time as $t = 0$.

6. It can be shown that when the decoder receives $1{,}000(1 + \zeta_{1{,}000})$ packets, where $\zeta_{1{,}000}$ is a positive number very close to zero, it can successfully decode all 1,000 input packets with high probability [35], [36].

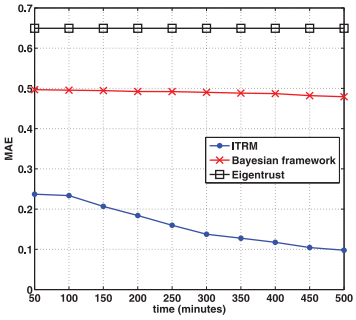


Fig. 13. MAE performance of various schemes for bad mouthing when $W = 0.30$.

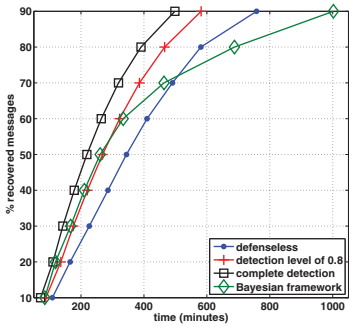


Fig. 14. Fraction of the recovered messages versus time for $W = 0.10$ with the LW mobility model.

mislead the detection mechanism. In addition to the attacks on the security mechanism (i.e., the trust management and the detection algorithms), malicious nodes mount attacks on the network communication protocol by both dropping the legitimate packets they have received from reliable nodes (with different packet drop rates) and generating their own flows to deliver to other (malicious) nodes via the legitimate nodes. The ultimate goal of the adversary is to degrade the network performance (i.e., data availability and packet delivery ratio).

Mean absolute error. In Fig. 13, we compared the performance of ITRM with the Bayesian reputation management framework in [10] and the EigenTrust algorithm [21] (in the DTN environment presented before) in terms of MAE when the fraction of the malicious raters (W) is 0.30. In other words, for each legitimate judge, we computed the average MAE (between the actual reputation value and the computed reputation value) based on the reputation values computed at that judge node. Further, since each legitimate judge node computes the reputation values (of the other nodes) itself using ITRM, Bayesian framework or EigenTrust, we computed the average MAE over all legitimate nodes.

From these simulation results, we conclude that ITRM significantly outperforms the Bayesian framework and the EigenTrust algorithm in the presence of attacks. Further, for different values of W and for different parameters in the LW mobility model, we still observed the superiority of ITRM over the other schemes. We note that since the Bayesian framework shows a better performance than the EigenTrust in terms of MAE, we compare the performance of ITRM with the Bayesian framework for data availability and packet delivery ratio in the rest of this section.

Availability. We define the *availability* as the percentage of recovered messages (by their final destinations) in the network at a given time. In Figs. 14 and 15, we showed the percentage of recovered messages versus time for the following scenarios:

1. when there is no defense against the malicious nodes and each malicious node has a packet drop rate of 1,
2. when a detection level of 0.8 is used by ITRM (in which each judge node is supposed to identify and isolate all the Byzantine nodes whose packet drop rates are 0.8 or higher),
3. when a complete detection is used by ITRM (in which all malicious nodes are supposed to be detected and isolated regardless of their packet drop rate), and
4. when the Bayesian reputation management framework in [10] is used to detect the malicious nodes. We note that in the second, third, and fourth scenarios, the packet drop rates by the malicious nodes are uniformly distributed between 0 and 1 in order to make the detection harder.

Further, in the second, third, and fourth scenarios, we assume the attack on the security mechanism as described before.

The plots show that the percentage of recovered messages at a given time significantly decreases with increasing W for the defenseless scheme. On the other hand, we observed a considerable improvement in the percentage of recovered messages even after a high-level detection ($p = 0.8$) using the proposed scheme. We further observed that the Bayesian reputation management framework in [10] fails to provide high data availability with low latency. This is due to the fact that when the malicious nodes collaboratively attack the reputation management scheme, reputation systems which rely on the Bayesian Approach (such as [10]) result in high MAE in the reputation values of the nodes (as illustrated in Fig. 13). Therefore, the reputation mechanism in [10] not only fails to detect all malicious nodes in the network, but it also labels some reliable nodes (which are victimized by the malicious nodes using the bad-mouthing attack) as malicious. Moreover, we considered the *reliable message delivery* as the probability of the delivery of a single specific message to its destination at any given time. Thus, the probability of recovery (of a specific message) at the destination node at

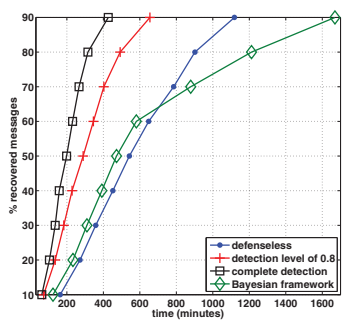


Fig. 15. Fraction of the recovered messages versus time for $W = 0.40$ with the LW mobility model.

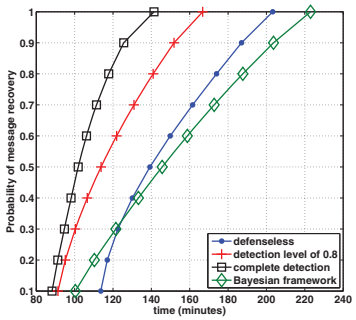


Fig. 16. Probability of message recovery for a single flow versus time for $W = 0.10$ with the LW mobility model.

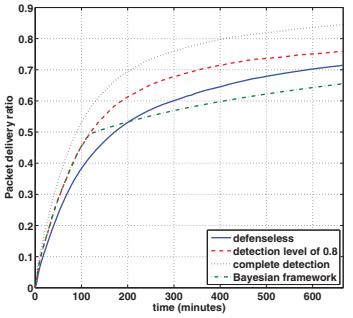


Fig. 18. Packet delivery ratio versus time for $W = 0.10$ with the LW mobility model.

any given time is plotted (while other flows still exist) in Figs. 16 and 17. These figures also illustrate the improvement in reliable message delivery as a result of the proposed scheme even after a high-level detection. We again observed that the reputation mechanism in [10] fails to provide fast reliable message delivery due to the vulnerability of the Bayesian reputation management framework to detect malicious nodes.

Comparing the time required for a high success probability (for detection) in Figs. 11 and 12 and the time required to have high data availability at the receivers, we observed that the ITRM enables the judge nodes to calculate the reputations of all the network nodes in a relatively short amount of time. In other words, the time required to calculate the reputation values of all the network nodes at a judge node is significantly less than the time required for the transmission of a single message, which is a significant result for DTNs. Further, the overhead caused by the extra messages between the nodes due to the security protocol is negligible when compared with the data packets. This is because the overhead due to the security mechanism is dominated by the signed receipts from the suspect nodes to prove the deliveries by the suspect nodes. As we mentioned before, knowing how long it takes to have a high success probability at a judge node for a given detection level (from the results in Figs. 11 and 12), the witnesses can determine the lifetimes of the signed receipts. For example, in the LW mobility model used, the scheme provides a high probability of success (S) in approximately 70 minutes. Therefore, the lifetime of a signed receipt is estimated as 70 minutes, on the average. Moreover, for the chosen mobility model, each node establishes (on the average) 30 contacts in 70 minutes. This means that a suspect node transfers approximately 30 signed receipts to a witness node upon its contact. Since the length of the signature is about 20 bytes [42] and the size of a data packet is 128 bytes, 30 signed receipts can be delivered via five data packets. Considering the data rates of 250 kbps, the overhead of five data packets becomes negligible when compared to the entire message exchange between two nodes during the contact. This also shows that the proposed algorithm does not introduce a significant overhead burden on the network.

Packet delivery ratio. We define the packet delivery ratio as the ratio of the number of legitimate packets received by their destinations to the number of legitimate packets transmitted by their sources. Therefore, we observed the impact of malicious nodes on the packet delivery ratio and the progress achieved as a result of our scheme in Figs. 18 and 19. As before, we consider

1. the defenseless scheme,
2. a detection level of 0.8,
3. a complete detection, and
4. the Bayesian reputation management framework in [10].

We observed a notable improvement in the packet delivery ratio as a result of the proposed scheme. As W increases, the packet delivery ratio of the defenseless scheme decreases significantly while our proposed scheme still provides a high packet delivery ratio even at the detection level of 0.8, which illustrates the robustness of the proposed scheme. Finally, we observed that the scheme in [10] fails to provide a high packet delivery ratio due to its vulnerability against colluding malicious nodes as discussed before.

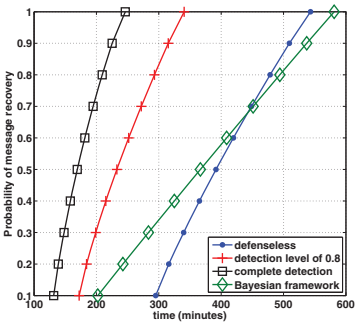


Fig. 17. Probability of message recovery for a single flow versus time for $W = 0.40$ with the LW mobility model.

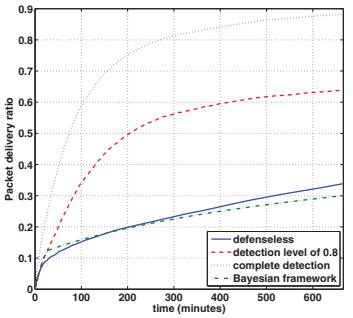


Fig. 19. Packet delivery ratio versus time for $W = 0.40$ with the LW mobility model.

4 Conclusion

In this paper, we introduced a robust and efficient security mechanism for delay-tolerant networks. The proposed security mechanism consists of a trust management mechanism and an iterative reputation management scheme. The trust management mechanism enables each network node to determine the trustworthiness of the nodes with which it had direct transactions. On the other hand, ITRM takes advantage of an iterative mechanism to detect and isolate the malicious nodes from the network in a short time. We studied the performance of the proposed scheme and showed that it effectively detects the malicious nodes even in the presence of the attacks on the trust and detection mechanisms. We also illustrated that the proposed scheme is far more effective than the Bayesian framework and EigenTrust in computing the reputation values in a DTN environment. Moreover, using computer simulations we showed that the proposed mechanism provides high data availability with low information latency by detecting and isolating the malicious nodes in a short time.

Acknowledgments

This material is based upon work supported by the US National Science Foundation under Grant No. IIS- 1115199, and a gift from the Cisco University Research Program Fund, an advised fund of Silicon Valley Community Foundation.

References

[1] K. Fall, "A Delay-Tolerant Network Architecture for Challenged Internets," *Proc. ACM SIGCOMM,* pp. 27-34, 2003.

[2] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-Coding Based Routing for Opportunistic Networks," *Proc. ACM SIGCOMM Workshop Delay-Tolerant Networking (WDTN '05),* pp. 229-236, 2005.

[3] Y. Zhu, D. Guo, and M.L. Honig, "A Message-Passing Approach for Joint Channel Estimation, Interference Mitigation and Decoding," *IEEE Trans. Wireless Comm.,* vol. 8, no. 12, pp. 6008-6018, Dec. 2009.

[4] H. Pishro-Nik and F. Fekri, "Results on Punctured Low-Density Parity-Check Codes and Improved Iterative Decoding Techniques," *IEEE Trans. Information Theory,* vol. 53, no. 2, pp. 599-614, Feb. 2007.

[5] E. Ayday, H. Lee, and F. Fekri, "An Iterative Algorithm for Trust and Reputation Management," *Proc. IEEE Int'l Symp. Information Theory (ISIT '09),* 2009.

[6] A.A. Pirzada, C. McDonald, and A. Datta, "Performance Comparison of Trust-Based Reactive Routing Protocols," *IEEE Trans. Mobile Computing,* vol. 5, no. 6, pp. 695-710, June 2006.

[7] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad-Hoc Networks," *Proc. ACM MobiCom,* pp. 255-265, 2000.

[8] K. Paul and D. Westhoff, "Context Aware Detection of Selfish Nodes in DSR Based Ad-Hoc Networks," *Proc. IEEE GlobeCom,* pp. 178-182, 2002.

[9] S. Buchegger and J. Boudec, "Performance Analysis of CONFIDANT Protocol (Coorperation of Nodes: Fairness in Dynamic Ad-Hoc Networks)," *Proc. ACM MobiHoc,* June 2002.

[10] S. Buchegger and J. Boudec, "A Robust Reputation System for P2P and Mobile Ad-Hoc Networks," *Proc. Second Workshop the Economics of Peer-to-Peer Systems,* 2004.

[11] E. Ayday and F. Fekri, "Using Node Accountability in Credential Based Routing for Mobile Ad-Hoc Networks," *Proc. Fifth IEEE Int'l Conf. Mobile Ad-Hoc and Sensor Systems,* 2008.

[12] E. Ayday and F. Fekri, "A Protocol for Data Availability in Mobile Ad-Hoc Networks in the Presence of Insider Attacks," *Elsevier Ad Hoc Networks,* vol. 8, no. 2, pp. 181-192, Mar. 2010.

[13] P. Dewan, P. Dasgupta, and A. Bhattacharya, "On Using Reputations in Ad-Hoc Networks to Counter Malicious Nodes," *Proc. 10th Int'l Conf. Parallel and Distributed Systems (ICPADS '04),* 2004.

[14] K. Liu, J. Deng, P.K. Varshney, and K. Balakrishnan, "An Acknowledgment-Based Approach for the Detection of Routing Misbehavior in MANETs," *IEEE Trans. Mobile Computing,* vol. 6, no. 5, pp. 536-550, May 2007.

[15] W. Yu and K.R. Liu, "Game Theoretic Analysis of Cooperation Stimulation and Security in Autonomous Mobile Ad-Hoc Networks," *IEEE Trans. Mobile Computing,* vol. 6, no. 5, pp. 507-521, May 2007.

[16] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara, "Reputation Systems: Facilitating Trust in Internet Interactions," *Comm. ACM,* vol. 43, no. 12, pp. 45-48, 2000.

[17] A. Jøsang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems,* vol. 43, no. 2, pp. 618-644, 2007.

[18] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," *Proc. 10th Int'l Conf. Information and Knowledge Management (CIKM '01),* pp. 310-317, 2001.

[19] F. Cornelli, E. Damiani, S.D.C. di Vimercati, S. Paraboschi, and P. Samarati, "Choosing Reputable Servents in a P2P Network," *Proc. 11th Int'l Conf. World Wide Web (WWW '02),* pp. 376-386, 2002.

[20] E. Damiani, D.C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," *Proc. Ninth ACM Conf. Computer and Comm. Security (CCS '02),* pp. 207-216, 2002.

[21] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks," *Proc. 12th Int'l Conf. World Wide Web (WWW '03),* pp. 640-651, 2003.

[22] C. Dellarocas, "Immunizing Online Reputation Reporting Systems against Unfair Ratings and Discriminatory Behavior," *Proc. Second ACM Conf. Electronic Commerce (EC '00),* pp. 150-157, 2000.

[23] P. Macnaughton-Smith, W.T. Williams, M.B. Dale, and L.G. Mockett, "Dissimilarity Analysis: A New Technique of Hierarchical Sub-Division," *Nature,* vol. 202, pp. 1034-1035, 1964.

[24] A. Seth and S. Keshav, "Practical Security for Disconnected Nodes," *Proc. First IEEE ICNP Workshop Secure Network Protocols (NPSec),* pp. 31-36, 2005.

[25] S. Cui, P. Duan, and C. Chan, "An Efficient Identity-Based Signature Scheme with Batch Verifications," *Proc. First Int'l Conf. Scalable Information Systems (InfoScale '06),* p. 22, 2006.

[26] A. Kate, G. Zaverucha, and U. Hengartner, "Anonymity and Security in Delay Tolerant Networks," *Proc. Third Int'l Conf. Security and Privacy in Comm. Networks (SecureComm '07),* 2007.

[27] J. Burgess, G. Bissias, M. Corner, and B. Levine, "Surviving Attacks on Disruption-Tolerant Networks without Authentication," *Proc. Eighth ACM Int'l Symp. Mobile Ad Hoc Networking and Computing,* pp. 61-70, 2007.

[28] Y. Yang, Q. Feng, Y.L. Sun, and Y. Dai, "RepTrap: A Novel Attack on Feedback-Based Reputation Systems," *Proc. Fourth Int'l Conf. Security and Privacy in Comm. Networks (SecureComm '08),* pp. 1-11, 2008.

[29] B.N. Vellambi and F. Fekri, "Results on the Improved Decoding Algorithm for Low-Density Parity-Check Codes over the Binary Erasure Channel," *IEEE Trans. Information Theory,* vol. 53, no. 4, pp. 1510-1520, Apr. 2007.

[30] G. Zacharia, A. Moukas, and P. Maes, "Collaborative Reputation Mechanisms in Electronic Marketplaces," *Proc. 32nd Ann. Hawaii Int'l Conf. System Sciences (HICSS '99),* vol. 8, 1999.

[31] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong, "On the Levy Walk Nature of Human Mobility," *Proc. IEEE INFOCOM,* 2008.

[32] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proc. ACM MobiCom,* pp. 85-97, 1998.

[33] A. Petz, J. Enderle, and C. Julien, "A Framework for Evaluating DTN Mobility Models," *Proc. Second Int'l Conf. Simulation Tools and Techniques,* pp. 94:1-94:8, 2009.

[34] S. Hong, I. Rhee, S.J. Kim, K. Lee, and S. Chong, "Routing Performance Analysis of Human-Driven Delay Tolerant Networks Using the Truncated Levy Walk Model," *Proc. First ACM SIGMOBILE Workshop Mobility Models,* pp. 25-32, 2008.

[35] M. Luby, "LT Codes," *Proc. 43rd Symp. Foundations of Computer Science (FOCS '02),* pp. 271-280, 2002.

[36] A. Shokrollahi, "Raptor Codes," *IEEE Trans. Information Theory,* vol. 52, no. 6, pp. 2551-2567, June 2006.

[37] B.N. Vellambi, R. Subramanian, F. Fekri, and M. Ammar, "Reliable and Efficient Message Delivery in Delay Tolerant Networks Using Rateless Codes," *Proc. First Int'l MobiSys Workshop Mobile Opportunistic Networking (MobiOpp '07)*, pp. 91-98, 2007.
[38] B.H. Bloom, "Space/Time Trade-Offs in Hash Coding with Allowable Errors," *ACM Comm.*, vol. 13, no. 7, pp. 422-426, July 1970.
[39] F. Hao, M. Kodialam, and T.V. Lakshman, "Building High Accuracy Bloom Filters Using Partitioned Hashing," *Proc. ACM Int'l Conf. Measurement and Modeling of Computer Systems*, pp. 277-288, 2007.
[40] R. Groenevelt, P. Nain, and G. Koole, "The Message Delay in Mobile Ad Hoc Networks," *Performance Evaluation*, vol. 62, nos. 1-4, pp. 210-228, 2005.
[41] A. Whitby, A. Josang, and J. Indulska, "Filtering Out Unfair Ratings in Bayesian Reputation Systems," *Proc. Seventh Int'l Workshop Trust in Agent Societies (AAMAS '04)*, 2004.
[42] C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen, "An Efficient Identity Based Batch Verification Scheme for Vehicular Sensor Networks," *Proc. IEEE INFOCOM*, 2008.

Erman Ayday received the BS degree in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey, in 2005. He received the MS and PhD degrees from the School of Electrical and Computer Engineering (ECE), Georgia Institute of Technology (Georgia Tech), Atlanta, in 2007 and 2011, respectively. He is now a postdoctoral researcher at Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland, in the Laboratory for Communications and Applications 1 (LCA1). His research interests include wireless network security, privacy, game theory for wireless networks, trust and reputation management, and recommender systems. He was the recipient of the 2010 Outstanding Research Award from the Center of Signal and Image Processing (CSIP) at Georgia Tech and the 2011 ECE Graduate Research Assistant (GRA) Excellence Award from Georgia Tech. He is a student member of the IEEE and the ACM.

Faramarz Fekri received the PhD degree from the Georgia Institute of Technology in 2000, and has since been with the faculty of the School of Electrical and Computer Engineering there, where he currently holds a full professor position. He serves on the editorial board of the *IEEE Transactions on Communications* and on the technical program committees of several IEEE conferences. His current research interests are in the area of communications and signal processing, in particular coding and information theory, information processing for wireless and sensor networks, and communication security. He received the US National Science Foundation CAREER Award in 2001, the Southern Center for Electrical Engineering Education (SCEEE) Research Initiation Award in 2003, and the Outstanding Young Faculty Award of the School of Electrical and Computer Engineering in 2006. He is a senior member of the IEEE.