Trust Management and Adversary Detection for Delay Tolerant Networks

Erman Ayday
School of Electrical and Comp. Eng.
Georgia Institute of Technology
Atlanta, GA 30332, USA
Email: erman@ece.gatech.edu

Hanseung Lee
School of Electrical and Comp. Eng.
Georgia Institute of Technology
Atlanta, GA 30332, USA
Email: hanseung.lee@ece.gatech.edu

Faramarz Fekri
School of Electrical and Comp. Eng.
Georgia Institute of Technology
Atlanta, GA 30332, USA
Email: fekri@ece.gatech.edu

***Abstract*—Delay Tolerant Networks (DTNs) have been identified as one of the key areas in the field of wireless communications. They 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 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, the techniques to build a trust mechanism for MANETs 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 developed an iterative malicious node detection mechanism for DTNs which is far more effective than existing techniques. Our results indicate the proposed scheme provides high data availability and packet-delivery ratio with low latency in DTNs under adversary attacks.**

I. Introduction

Delay Tolerant Networks (DTNs) are 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. In contrast, DTNs are characterized by intermittent contacts between nodes. 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 towards their destinations. Therefore, DTNs are characterized by large end-to-end latency, opportunistic communication over intermittent links, error-prone links, and (most importantly) the lack of end-to-end path from a source to its destination. By the above discussion, it can be argued that MANETs are a special class of DTNs.

As in MANETs, adversary may mount several threats against DTNs to reduce the network performance. The most serious attacks are due to the Byzantine (insider) adversary. A Byzantine adversary (i.e., a physically captured and controlled legitimate node) can do serious damage to the network in terms of data availability, latency, and throughput. Optimal attacks for a Byzantine node are to: 1. Drop (or modify) the legitimate packets it receives from legitimate nodes, and 2. Inject its own packets to use the network resources and deny the network operation.

In MANETs, reputation-based trust management systems are shown to be an effective way to cope with adversary. 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 impractical in such networks. Common techniques to build reputation in MANETs are based on the direct measurements of a suspicious node [2], acknowledgement (ACK) messages from the destination [3], and indirect measurements of the suspicious nodes [4]. However, these techniques are not applicable in DTNs due to opportunistic communication during contact times, the lack of feedback, and the lack of end-to-end paths.

Our main objective in this paper is to develop a security mechanism for DTNs which enables us to detect misbehavior due to Byzantine adversaries. 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 global reputation systems stems from the prior success in the use of iterative algorithms, such as message passing techniques [5] in the decoding of Low-Density Parity-Check (LDPC) codes in erasure channels. We believe that the significant benefits offered by iterative algorithms can be tapped in to benefit the field of global reputation systems. To achieve this, we proposed the Iterative Trust and Reputation Mechanism (ITRM) [6], and in this work, we explore its application on DTNs. We develop a distributed malicious node detection mechanism for DTNs using ITRM which enables every node to evaluate other nodes based on their past behavior. 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 as follows.

1) We propose a novel iterative scheme for trust management and malicious node detection.
2) The overall scheme provides high data availability 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 has a computational complexity that is linear with the number of nodes in the network. 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 II, we briefly describe ITRM. Then, in Section III, we present the application of ITRM to DTNs and the proposed security mechanism in detail. Moreover, we evaluate the proposed scheme via computer simulations. Finally, in Section IV, we conclude the paper.

A. Related Work

Several works in the literature have focused on securing DTNs. In [7], the challenges of providing secure communication (i.e., confidentiality) in DTNs is discussed and the use of IBC [8] is suggested. In [9], source authentication and anonymous communication as well as message confidentiality are provided using IBC. We note that the existing techniques to secure DTNs

This material is based upon work supported partially by the Army Research Office (ARO) under grant 49586CI.

are aimed to provide data confidentiality and authentication only. On the other hand, our proposed scheme provides malicious node detection and high data availability with low latency.

The most famous and primitive global reputation system is the one that is used in eBay. Other well-known websites such as Amazon, Epinions, and AllExperts use more advanced reputation mechanisms than eBay. Their reputation mechanisms compute the average (or weighted average) of the received ratings to evaluate the global reputation of a product (or a peer) [10]. Hence, these schemes are vulnerable to collaborative attacks by malicious peers. Use of the Bayesian Approach is also proposed in [11]. Finally, [12] proposed to use the *Cluster Filtering* method [13] for reputation. Different from the existing schemes, ITRM algorithm [6] is a graph based iterative algorithm motivated by the previous success on iterative message passing techniques.

II. Iterative Trust and Reputation Management Mechanism (ITRM)

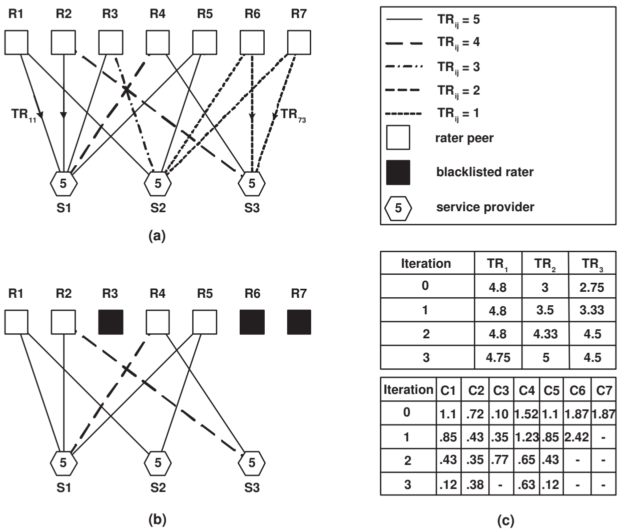
We initially describe ITRM [6] and its security evaluation briefly. Then, we will utilize it for DTNs. As in every trust and reputation management mechanism, we have two main goals: 1. Computing the service quality (reputation) of the peers who provide a service (henceforth referred to as Service Providers 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 analyzing their feedback about SPs. We assume two different sets in the system: i) The set of SPs, and ii) The set of service consumers. Transactions occur between SPs and consumers, and consumers provide feedbacks in the form of ratings about SPs after each transaction. We denote the global reputation of the j^{th} SP and the rating that the rater i reports about the SP j as TR_j and TR_{ij}, respectively. Moreover, we let R_i denote the (rating) trustworthiness of the i^{th} rater. We consider the following major attacks that are common for any trust and reputation management mechanisms: i) 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 ii) Ballot-stuffing, in which malicious raters collude to increase the reputation values of peers with low reputations.

The first step of ITRM is to interpret the collection of the raters and the SPs together with their associated relations as a bipartite graph, as in Fig. 1(a). In this representation, each rater corresponds to a *check vertex* in the graph, shown as a square. Further, each SP is represented by a *bit vertex* shown as a hexagon in the graph. Furthermore, each rating is represented by a directed edge from the check-vertex to the bit-vertex.

In ITRM, prior to the start of the iterations, we compute the initial value of each bit-vertex j based on the weighted average of the edge values incident to it. Equivalently, we compute

$$TR_j = \frac{\sum_{i \in A} R_i \times TR_{ij}}{\sum_{i \in A} R_i}, \tag{1}$$

where A is the set of check-vertices connected to the bit-vertex j. It is interesting to note that the initial values resemble the received information from the channel in the channel coding problem. Then, the first iteration starts. We first compute the average inconsistency factor C_i of each check-vertex i using the values of the bit-vertices that it is connected to. That is, we compute $C_i = [1/|\Upsilon|]\sum_{j \in \Upsilon} d(TR_{ij}, TR_j)$ where Υ 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. 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. Once the check-vertex i is blacklisted, we delete its ratings (TR_{ij}) for all the bit-vertices j it is connected to. Then, we update the values of all the bit-vertices using (1). The iterative procedure proceeds to other rounds exactly in the same way as the first round. We stop the iterations when the inconsistencies of all check-vertices (excluding the blacklisted ones) fall below τ and update the R_i values of the raters based on a Beta distribution [14].



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	-	-

Fig. 1: Illustrative example of ITRM.

As an example, ITRM is illustrated in Fig. 1 for 7 raters, 3 SPs, and $\tau = 0.7$. It is assumed that the ratings are integers from 1 to 5 and the actual reputations, $\hat{TR_j}$, are equal to 5. Further, we use the $\mathcal{L}^1$ norm (absolute value) as the distance metric to calculate the inconsistencies of the raters. For simplicity, we assumed R_i's to be equal for all raters. Furthermore, we assumed that the raters 1, 2, 3, 4, and 5 are reliable, but 6 and 7 are malicious. The malicious raters are mounting the "bad-mouthing" attack by rating the same set of SPs as one. Fig. 1(a) shows the TR_{ij} values (illustrated by different line-styles) prior to the execution of the algorithm. The TR_j values and the individual inconsistencies of the raters after each iteration are also illustrated in Fig. 1(c). We note that the algorithm stops at the third iteration when all the raters have inconsistencies less than τ. Fig. 1(c) indicates how the iterative algorithm gives better estimates of TR_j's compared to the weighted averaging method (which corresponds to iteration zero). Fig. 1(b) illustrates the graph after the final iteration and shows that the malicious raters (6 and 7) are blacklisted and isolated. Moreover, rater 3, although reliable, is also blacklisted at the third iteration. We note that this situation is possible when a reliable but faulty rater's ratings have a large deviation from the other reliable raters.

A. Security Evaluation of ITRM

Here, we briefly discuss the security evaluation of ITRM [6]. We let $\hat{TR_j}$ be the actual reputation value of the j^{th} SP. We assumed that the ratings are integers from 1 to 5, rating given by a reliable rater is a random variable with folded normal distribution (mean $\hat{TR_j}$ and variance 0.5), and the number of newly generated ratings, per time-slot, by a reliable rater is a random variable with Yule-Simon distribution (with $\rho = 1$), which is commonly used in modeling online systems. We further assumed that the malicious raters initiate bad-mouthing. Moreover, they collude and attack the same set Γ of SPs at each time-slot. Hence, at each time-slot, the malicious raters choose the SPs from Γ (whose actual reputation values, $\hat{TR_j}$, are 5) and rate them as 4. The malicious raters do not deviate very much from the actual reputation values to remain undercover as many time-slots as possible. We used the following parameters for the rest of this section to illustrate our results: 200 raters, 100 SPs, $|\Gamma| = 5$ and $\tau = 0.4$ (the choice of τ is based on analytical results in [6]).

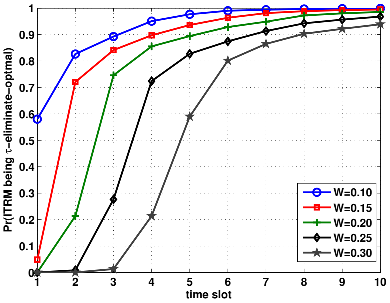


Fig. 2: Probability that ITRM is a τ-eliminate-optimal scheme versus fraction of malicious raters for $\tau = 0.4$.

A 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 the averaging scheme) exceeds the threshold τ. In Fig. 2, we illustrated the probability for ITRM to be a τ-eliminate-optimal scheme based on analytical results. We designed the scheme to tolerate up to 30% malicious raters ($W = 0.3$). As shown in Fig. 2, for W lower than 0.3, the waiting time becomes shorter to have a τ-eliminate-optimal scheme with a high probability.

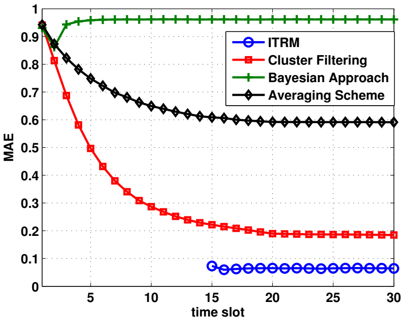


Fig. 3: MAE performance of various schemes for bad-mouthing when $W = 0.30$.

We further compared the performance of ITRM with three well-known and commonly used reputation management schemes: 1) *The Averaging Scheme*, 2) *Bayesian Approach*, and 3) *Cluster Filtering*. We obtained the performance of ITRM, at each time-slot, as the mean absolute error (MAE) $|TR_j - \hat{TR}_j|$, averaged over all the SPs that are under attack. Figure 3 illustrates the comparison of ITRM with the other schemes for the bad-mouthing attack when the fraction of malicious raters (W) is 0.3. The lags in the plot of ITRM correspond to waiting times to execute ITRM, computed based on our analytical results presented in Fig. 2. On the other hand, we executed the other three schemes starting from the first time-slot, since we observed that their performances were better that way. From these results, we concluded that ITRM significantly outperforms the Averaging Scheme and the Bayesian Approach in the presence of attacks. We identified that the reputation management scheme with the closest performance to ITRM is Cluster Filtering. However, the computational complexity of Cluster Filtering is much higher than ITRM. We calculated that Cluster Filtering introduces quadratic complexity while the computational complexity of ITRM is linear with the number of raters. As a result, ITRM is more scalable and suitable for large scale systems.

III. Trust Management and Adversary Detection in DTNs

A. *Adversary Models and Security Threats*

As discussed in Section I, we consider the challenging problem of countering Byzantine attacks. We note that the security issues such as source authentication and data authentication have been considered for disconnected networks in [7], [9]. Hence, they are not explicitly considered in this paper.

Packet drop and packet injection attack: 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: 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 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.

Random attack on trust management: 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.

Collaborative bad-mouthing (ballot-stuffing) on the detection scheme: 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 pass their tables to a legitimate node, they may collaboratively victimize the legitimate nodes (i.e., bad-mouthing) or help their malicious allies (i.e., ballot-stuffing) in their rating-table entries. This effectively reduces the detection performance of the system.

B. *Network/Communication Model*

Mobility model: We use a *contact-based* mobility model for our study [15], where the contact time between two nodes is modeled as a random variable with exponential distribution.

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

Routing and packet exchange protocol: We assume that messages at the source are packetized. Further, the source never transmits multiple copies of the same packet. We assume only single-copy routing since reliable single-copy routing with packetization is achieved by encoding the data packets using rateless codes [16] at the source node. The use of rateless coding improves reliability and latency in DTNs even when there is no adversary. Furthermore, exchange of packets between two nodes follows a back-pressure policy. To illustrate this, assume node 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. The nodes in contact pursue this policy for all different flows they have in their buffer. 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 needs to transfer some packets belonging to flow f to node B based on the back-pressure policy. In this situation, node A must fairly select the packets based on their previous hops (which is available via the packet format). That is, 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.

C. *Iterative Detection for DTNs*

In this section, we will describe how ITRM is adapted in DTNs as an iterative malicious node detection mechanism in which every node is able to decide on the faithfulness of all the nodes in the network. We will pick an arbitrary node in the network and present the algorithm from its point of view. 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 I), a judge node creates its own rating about another network node by collecting indirect measurements (feedbacks) 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 III-D) 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 reputations of all the nodes in the network in a timely manner, mitigating the effects of malicious nodes immediately. To achieve this, 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 considered as 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 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 II; which includes all the network nodes as bit-vertices. In other words, 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 non-empty entries. Hence, the resulting graph would have k check-vertices (the k^{th} 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 via simulations in Section III-E. Hence, when two nodes establish a contact, they exchange their rating tables. Once a judge node collects sufficient number of tables each with sufficient number of non-empty 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 ratings (feedbacks) 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. Moreover, in this work, we set all R_i values to one for simplicity.

D. 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 [17] and ID-based signature (IBS) [8]. 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. We note 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 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 [7], [9] 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. We will describe this scheme by using a toy example between 3 nodes A, B, and C. We denote the node that is evaluating as the *judge* (A), the node that is being evaluated as the *suspect* (B), and the direct contact of the suspect as the *witness* (C). The basic working principle of the mechanism is that after the judge has a transaction (in the form of passing packets) with a suspect, the judge waits to make contacts and receive feedback about the suspect from every node that has been in direct contact (witnesses) with the suspect.

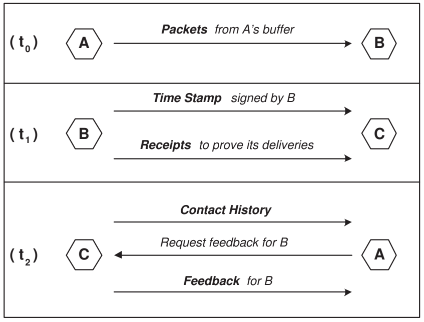


Fig. 4: Feedback mechanism between nodes A (judge), B (suspect) and C (witness).

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 $t_0 < t_1 < t_2$. Feedback mechanism between nodes A, B and C is illustrated in Fig. 4. At time t_0, A and B execute mutual packet exchange as described in Section III-B. 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 as we discussed in Section III-B (the fairness is measured from C's point of view). Moreover, node B transfers the receipts it received thus far to the witness C. Those receipts include the proofs of B's deliveries thus far and are signed by the nodes to which the packets were delivered. We note that the receipts expire in time and they are deleted from the buffers of the witnesses. Hence, they are not accumulated in the buffers of the nodes. 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 duration. Finally, when the judge 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 3 parts: i) Those receipts of B that are useful for A's evaluation, ii) 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 (C finds out A's packets by examining the headers as explained in Section III-B), and iii) C gives its own vote (in the form of *positive evaluation* as 1 or *negative evaluation* as 0) about B. C's vote depends on whether B followed the packet delivery policy explained in Section III-B.

After getting the feedback, the judge makes its evaluation for the suspect for this particular transaction. In other words, if A believes that the suspect node B delivered its packets following the protocol, it makes a “positive evaluation” as 1. Otherwise, the evaluation will be “negative” as 0. Each judge node uses the Beta distribution [14] to aggregate multiple evaluations it has made about a suspect using the associated feedbacks. The collection of multiple feedbacks generates the rating (verdict) of a judge node for a suspect node. We note that the feedbacks from the witnesses are not trustable. Because of the bad-mouthing and random attacks (discussed in Section III-A), 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 III-E.

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 nodes. This is unrealistic as the nodes may apply random attacks as in Section III-A. 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, a node with a higher packet drop rate 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) and the level of confidence is determined by the detection strategy. 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 confidence on a verdict. 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 is also good for use in a high detection level, 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.

E. Security Evaluation

In this section, we illustrate our simulation results for the metrics of interest. We also show the performance of the proposed scheme for malicious node detection, availability and packet delivery ratio via simulations (conducted using MATLAB).

Confidence on a Verdict: We assumed the mobility model of Section III-B with N nodes in the network. We studied the effect of random attack on the required number of feedbacks for a network with $N = 100$. We denote the percentage of the Byzantine nodes in the network as W. Figure 5 illustrates the variation of a (judge) node's confidence C 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.

Detection Performance: We illustrated the waiting time of a judge node before executing ITRM and evaluated the effects of collaborative attacks on the detection scheme for a network of size N in which the inter-contact time between two particular nodes is λ_i. 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 minimum number of non-empty 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. It is worth noting that we did not compare ITRM with other trust management schemes such as Weighted Average [4], Bayesian Approach [11], or Similarity Testing [18] because of the following reasons: 1. Since the judge node does not have any previous knowledge about the witness nodes and it trusts each witness node equally, weighted averaging is not applicable. 2. The purpose of the detection algorithm for the judge node is to determine the types of the nodes for which there are not sufficient feedbacks to have an entry in its own rating table. In this case Bayesian Approach becomes exactly the same as the Voting Technique (averaging the entries from the collected tables by weighting them equally). 3. Using a similarity test at the judge is vulnerable for DTNs because of the sparse rating tables. A malicious node can easily acquire good credibility in the eyes of a legitimate node by building its table so as to have a few common entries with the legitimate node's table.

We defined the *success* of a scheme as its capability of detecting all malicious nodes in the network. In Figs. 6 and 7, we illustrated the probability of success, S, of ITRM and the Voting Technique for different (k, s) pairs, and showed the time needed to obtain such a success probability. In our simulations, the size of the network (N) is 100, inter-contact time between two particular nodes (λ_i) is $1/500$, and the contact duration of two nodes is an exponentially distributed random variable with rate $\lambda_c = 3$. Whenever two nodes establish a contact, a transaction occurs between them in the form of packet exchange. Further, each judge node starts generating its rating table and each malicious node starts mounting its attack at time $t = 0$. We provided the evaluation for the collaborative bad-mouthing on the detection scheme only, as similar results hold for ballot-stuffing. In both figures, time is measured starting from $t = 0$. We note that these results also indicate the false positive (labeling a reliable node as malicious) and false negative (labeling a malicious node as reliable) probabilities of the proposed scheme as well. In other words, false positive and false negative probabilities are high when the probability of success (S) is low in Figs. 6 and 7. These results provide a good insight for a judge node about the instant it should apply ITRM. Based on our simulation results, we conclude that ITRM provides higher success rate in shorter time 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. 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.

For the rest of this section, we will illustrate our simulation results for different network parameters and show the performance of the proposed scheme for data availability and packet delivery ratio. We note that we did not compare the proposed scheme with existing DTN security schemes such as [9] since none of the existing schemes is aimed to provide data availability and malicious node detection as we did in this work. In all simulations, we assumed $N = 100$ nodes in the network, inter-contact time between two particular nodes (λ_i) is $1/500$, and the contact duration of two nodes is an exponentially distributed random variable with rate $\lambda_c = 3$. Moreover, the number of packets delivered during one time unit of the contact is 100. We assumed that a definite amount of time (2000 time-units) has passed since the lunch of the system and during this initialization period, nodes kept generating new messages and sending them to their destinations following a Poisson distribution with rate $\lambda_m = 1/100$. Further, during the initialization period, rating tables were being created at the judge nodes and attackers were mounting their attacks. Then, at time $t = 0$ (after the initialization period), we let the legitimate nodes start new flows to their destinations (while the previous flows still exist). Therefore, at time $t = 0$, we assumed each legitimate source node has 1000 information packets which are encoded via a rateless code for transmission. Hence, the number of encoded packets required by each destination to recover a message is roughly 1000 [16]. We evaluated the data availability and packet delivery ratio 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. We ran each simulation 100 times to get an average. We note that there is no motive to select these simulation parameters. We simulated the proposed scheme with different simulation parameters (for different network and mobility models) and obtained similar trends.

Availability and Packet Delivery Ratio: We defined the availability as the percentage of recovered messages (by their final destinations) in the network at a given time. We note that there is no other trust management scheme for disconnected networks to compare our scheme with. Moreover, due to the reasons described in Section I, we only compared the proposed scheme with the defenseless scheme. In Figs. 8 and 9, we showed the percentage of recovered messages versus time for the following scenarios: i) There is no defense against the malicious nodes and each malicious node has a packet drop rate of 1, ii) A detection level of 0.8 (in which each judge node identifies and isolates all the Byzantine nodes whose packet drop rates are

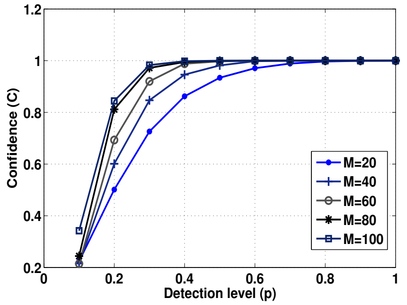


Fig. 5: Confidence of a judge node on its verdict vs. the detection level for $W = 10\%$.

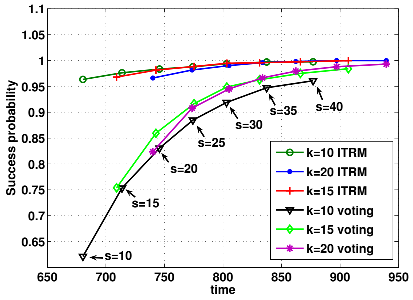


Fig. 6: Probability of detection success for fixed k and varying s values.

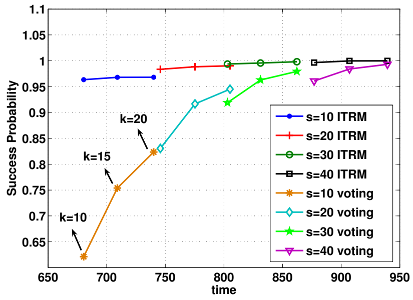


Fig. 7: Probability of detection success for fixed s and varying k values.

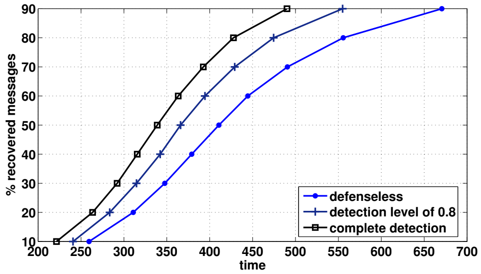


Fig. 8: Fraction of the recovered messages versus time for $W = 10\%$.

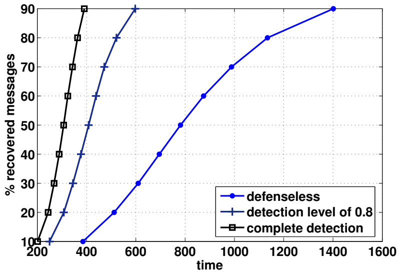


Fig. 9: Fraction of the recovered messages versus time for $W = 40\%$.

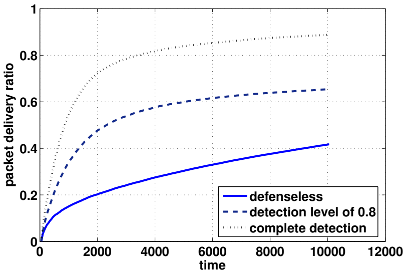


Fig. 10: Packet delivery ratio versus time for $W = 40\%$.

0.8 or higher), and iii) A complete detection (in which all malicious nodes are detected and isolated regardless of their packet drop rate). We note that in the second scenario, malicious nodes have packet drop rates uniformly distributed between 0 and 1 because the proposed trust management scheme forces them to drop legitimate packets with lower rates rather than consistently dropping. 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 at a given time even after a high level detection ($p = 0.8$) using the proposed scheme.

We defined 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 Fig. 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.

IV. 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 trust and reputation mechanism (ITRM). The trust management mechanism enables each network node to determine the trustworthiness of the nodes that it had a direct transaction. 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 voting-based techniques in detecting Byzantine nodes. Moreover, using computer simulations we showed that the proposed mechanism provides high data availability with low latency by detecting and isolating the malicious nodes.

References

[1] K. Fall, “A delay-tolerant network architecture for challenged internets,” *ACM SIGCOMM*, pp. 27–34, 2003.

[2] S. Marti, T. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad-hoc networks,” *Proceedings of ACM International Conference on International Conference on Mobile Computing and Networking (MobiCom00)*, pp. 255–265, 2000.

[3] 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.

[4] S. Buchegger and J. Boudec, “Performance analysis of CONFIDANT protocol (coorperation of nodes: Fairness in dynamic ad-hoc networks),” *Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Jun. 2002.

[5] B. F. F.R. Kschischang and H. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on Information Theory*, 2001.

[6] E. Ayday, H. Lee, and F. Fekri, “An iterative algorithm for trust and reputation management,” *ISIT ’09: Proceedings of IEEE International Symposium on Information Theory*, 2009.

[7] A. Seth and S. Keshav, “Practical security for disconnected nodes,” *Proceedings of the 1st IEEE ICNP Workshop on Secure Network Protocols (NPSec)*, pp. 31–36, 2005.

[8] S. Cui, P. Duan, and C. Chan, “An efficient identity-based signature scheme with batch verifications,” *Proceedings of the 1st International Conference on Scalable Information Systems (InfoScale ’06)*, p. 22, 2006.

[9] A. Kate, G. Zaverucha, and U. Hengartner, “Anonymity and security in delay tolerant networks,” *Proceedings of the 3rd International Conference on Security and Privacy in Communication Networks*, 2007.

[10] G. Zacharia, A. Moukas, and P. Maes, “Collaborative reputation mechanisms in electronic marketplaces,” *Proceedings of the Thirty-second Annual Hawaii International Conference on System Sciences-Volume 8*, 1999.

[11] S. Buchegger and J.Boudec, “Coping with false accusations in misbehavior reputation systems for mobile ad-hoc networks,” *EPFL-DI-ICA Technical Report IC/2003/31*, 2003.

[12] C. Dellarocas, “Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior,” *EC ’00: Proceedings of the 2nd ACM conference on Electronic commerce*, pp. 150–157, 2000.

[13] P. Macnaughton-Smith, W. T. Williams, M. B. Dale, and L. G. Mockett, “Dissimilarity analysis: A new technique of hierarchical sub-division,” *Natue(202)*, pp. 1034–1035, 1964.

[14] S. Buchegger and J. Boudec, “The effect of rumor spreading in reputation systems for mobile ad-hoc networks,” *Proc. Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt ’03)*, 2003.

[15] A. Khelil, P. J. Marrón, and K. Rothermel, “Contact-based mobility metrics for delay-tolerant ad-hoc networking.” pp. 435–444, 27-29 Sept 2005.

[16] A. Shokrollahi, “Raptor codes,” *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, June 2006.

[17] B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” *ACM Communications*, vol. 13, no. 7, pp. 422–426, Jul. 1970.

[18] M. Srivatsa, L. Xiong, and L. Liu, “Trustguard: countering vulnerabilities in reputation management for decentralized overlay networks,” *Proceedings of the International Conference on World Wide Web*, pp. 422–431, 2005.