Wireless Netw
DOI 10.1007/s11276-016-1320-1

Design of efficient lightweight strategies to combat DoS attack in delay tolerant network routing

Sujoy Saha[1] · Subrata Nandi[1] · Rohit Verma[2] · Satadal Sengupta[2] · Kartikeya Singh[1] · Vivek Sinha[1] · Sajal K. Das[3]

Abstract Delay tolerant networks (DTNs) are characterized by delay and intermittent connectivity. Satisfactory network functioning in a DTN relies heavily on co-ordination among participating nodes. However, in practice, such co-ordination cannot be taken for granted due to possible misbehaviour by relay nodes. Routing in a DTN is, therefore, vulnerable to various attacks, which adversely affect network performance. Several strategies have been proposed in the literature to alleviate such vulnerabilities—they vary widely in terms of throughput, detection time, overhead etc. One key challenge is to arrive at a tradeoff between detection time and overhead. We observe that the existing table-based reactive strategies to combat Denial-of-service (DoS) attacks in DTN suffer from two major drawbacks: high overhead and slow detection. In this paper, we propose three secure, light-weight and time-efficient routing algorithms for detecting DoS attacks (Blackhole and Grey-hole attacks) in the Spray & Focus routing protocol. The proposed algorithms are based on use of a small fraction of privileged (trusted) nodes. The first strategy, called TN, outperforms the existing table-based strategy with 20–30 % lesser detection time, 20–25 % higher malicious node detection and negligible overhead. The other two strategies, CTN_MI and CTN_RF explore the novel idea that trusted nodes are able to utilize each others' information/experience using their long range connectivity as and when available. Simulations performed using an enhanced ONE simulator reveals that investing in enabling connectivity among trusted nodes (as in CTN_RF) can have significant performance benefits.

Keywords Delay tolerant network (DTN) · Security · Routing · Denial-of-service (DoS) · Greyhole attack · Spray & Focus · Maliciousness · Trusted node

✉ Sujoy Saha
sujoy.ju@gmail.com

Subrata Nandi
subrata.nandi@gmail.com

Rohit Verma
rohitverma.kgp@gmail.com

Satadal Sengupta
satadal.sengupta.nit@gmail.com

Kartikeya Singh
kars007bond@gmail.com

Vivek Sinha
viveksinha356@gmail.com

Sajal K. Das
sdas@mst.edu

[1] National Institute of Technology Durgapur, Durgapur, India

[2] Indian Institute of Technology Kharagpur, Kharagpur, India

[3] Missouri University of Science and Technology, Rolla, MO, USA

1 Introduction

Network security has been an important area of research since the rise in popularity of the Internet. Numerous research efforts have addressed various security issues in traditional wired and wireless networks [13, 26, 27]. Networks have evolved considerably over the past few decades; while the base security issues remain the same, every new paradigm brings forward its own set of novel challenges. Mobile ad hoc networks (MANETs) had been introduced about a decade back, and a large amount of research effort has focussed on the security challenges in MANETs. While MANETs work well in environments where device mobility is constrained, the same cannot be

Published online: 29 June 2016

said about wireless ad hoc systems with sparse and intermittent connectivity.

MANETs are able to perform message transmission without fixed network infrastructure; however, in many practical network scenarios, such as vehicular communication, space/deep-sea communication, battlefield, communication in disaster affected regions [19] etc., such networks fail to ensure end-to-end connectivity due to low node density and intermittent disruptions. This situation limits the applicability of traditional ad hoc routing approaches, which tend to treat outages as failures and seek to find existing end-to-end paths. To deal with such limitations, the concept of Disruption Tolerant Network [6, 25] (DTN) was introduced.

DTNs use store-and-forward [23] approach, where data is incrementally moved and stored throughout the network, with a hope that it will eventually reach its destination. The packet is stored in the buffer of a node if it does not find the next-hop node *en route* to the destination. When a node encounters some other node, it decides whether to transmit the stored packet based on some predefined metric, such that delivery time (latency) is minimized and delivery probability is maximized. The metrics may include a function of previous encounters, last encounter time, utility value (determined using some predefined utility function), etc. Several metric-based routing protocols have been proposed in DTN, e.g., Spray & Focus [22], PRoPHET [12], MaxProp [2], etc.

DTNs have since gained immense popularity, and the possibility of using DTNs in challenged environments led to concentrated research efforts in its direction. More advanced DTN routing algorithms were developed, such as a *backpressure*-based routing protocol [4], which utilizes data packet scheduling, random walks and link estimation to decide forwarding paths in absence of end-to-end routes. Vehicular delay tolerant networks (VDTNs) were also conceived, where works such as by Zeng et al. [28], which proposed an energy-efficient communication scheme by using Nash Q-learning approach, came out as important contributions.

DTN related research has been identified as potential enablers for the Internet of Things (IoT) as well. Q. Zing et. al. [8] analyzed cross-layer heterogeneous integration issues and security issues in detail for each layer in IoT separately, and explored novel problems and solutions. Even in case of challenged networks, some pockets in the networks may allow the use of software defined network (SDN) kind of services, in conjunction with DTN. Security issues in such cases should also be considered therefore, as was studied in [14, 21].

Deeper research in the field of DTNs led to the exposure of security threats unique to the DTN environment. DTN and VDTN routing algorithms are designed with the basic assumption that nodes co-operate with each other in storing and forwarding packets. However, individual nodes may misbehave (indulge in malicious activities) or act selfishly, thus violating the established norm. Therefore, a DTN is vulnerable to various attacks and internal compromises, such as Intrusion [3], DoS [17], Greyhole [7], Tailgating [10], Blackhole [20], Wormhole [18], Misrouting [17, 20], Insider Attack [18], Layer Injection & removal [15, 30]. These may significantly affect the network performance, i.e., throughput, delivery probability, latency, etc.

Strategies proposed to combat malicious behaviour in DTN can be classified broadly into 3 categories, namely, *proactive approach*, which provides incentives to encourage co-operative behaviour [29], *user-interest oriented approach*, where rational behaviour of users based on personal interests or social ties are considered as design constraints and strategies are developed accordingly, and *reactive approach*, which tries to mitigate the effects of malicious activities by identifying their source and taking prohibitive action.

The *reactive approach* towards combating maliciousness works in two phases: the first phase involves detection of malicious nodes, while the second involves taking prohibitive action [11, 15, 16, 30]. The basic DTN routing algorithm is modified by appending additional fields to the routing table of each DTN node. Upon an encounter, apart from forwarding data packets, nodes also exchange partial/full information of their routing tables and execute the detection algorithm. The main challenge is to address the tradeoff between overhead (exchange and computational) and detection time.

Motivation Some of the existing works [3, 17, 20] design *reactive* combat strategies by deploying a small fraction of nodes as *trusted* nodes. Such trusted nodes are privileged with additional capabilities as compared to regular nodes. These nodes do not participate in data forwarding but only play the role of detecting a malicious node. Upon encounter, any regular node will transfer information (as specified by the secure routing algorithm) to the trusted node which then executes a detection algorithm to decide whether the node in contact is malicious. However, reports reveal that the existing strategies suffer from two major drawbacks—high overhead and slow detection.

In SRSnF [20], to combat Denial of Service (DoS) attack in Spray & Focus routing, all regular nodes participate in detection and are required to exchange several tables, namely, encounter table, transitivity table, and utility table, which incurs huge communication and computation overhead. In [3, 17], use of a small fraction of trusted nodes (termed *Ferries*) takes significantly longer time to detect the set of malicious nodes, which may be practically unacceptable in different application scenarios.

Both [3] and [17] model DoS attack in Prophet routing. Hence, an algorithm with relatively low overhead, and with capability of fast detection, is highly desirable but currently missing.

Objective This paper aims at designing practically useful secured routing algorithms which are light-weight and sufficiently fast in detecting malicious nodes. The proposed algorithms use specialized trusted nodes, however, in contrast to existing works, here we assume that trusted nodes are able to utilize each others' information/experience using their long range connectivity. The intuition behind the assumption comes from the following observation. Let's take the case of a large-scale natural disaster where there is partial/complete damage of conventional communication infrastructure in the affected area. However, there may be a few isolated small pockets (areas) where long distance connectivity (cellular or long distance WiFi) are still available. Moreover, as trusted nodes are privileged nodes, they may be enabled with satellite interface. Hence, even in extremely challenged scenarios, it is sometimes viable for trusted nodes to exchange information among themselves. The exchange is possible via some central database, by exploiting the long range links (whenever and wherever available) of these nodes. *The broad objective of this work is to study the impact of such connectivity among trusted nodes in designing practically useful secured DTN routing algorithms.* Here we consider Spray & Focus as a representative routing strategy.

Issues Our main task in this work is to model the DoS threat on Spray & Focus routing and make suitable modifications in the routing protocol. In order to deal with the tradeoff between computation and communication overhead versus performance benefits, specifically two major issues need to be addressed: (1) determine *what* information trusted nodes should exchange among themselves—they may exchange only the list of malicious nodes detected or several fields of the various tables maintained in nodes, and (2) determine *how often* they should exchange information—may be periodically or only when a new malicious node is detected.

Contribution Considering the aforementioned possibilities, we propose three different strategies to combat DoS attack in Spray & Focus. Performance comparisons are carried out using extensive simulations. The strategies and their performance can be summarized as follows:

1. Strategy 1: Design of a secured routing strategy using trusted nodes, denoted as (TN). The major enhancement of TN over SRSnF [20] is to enable detection by exchanging merely the encounter information instead of multiple raw tables. Results show that TN enables detection of 10–20 % more malicious nodes with negligibly higher overhead.
2. Strategy 2: Design of a secured routing strategy using connected trusted nodes (CTN), where trusted nodes exchange only Malicious nodes' Information List (MIL), denoted as (CTN_MI). Results show that CTN_MI diminishes detection time by 26 % and overhead by 6 %, and enables detection of 15–25 % more malicious nodes as compared to TN.
3. Strategy 3: Design of a secured routing strategy using connected trusted nodes (CTN), where trusted nodes exchange additional tables along with MIL, denoted as (CTN_RF). In this approach, cent percent malicious nodes are detected in shortest path movement model. Results also show that CTN_RF reduces detection time by 45 % over CTN_MI, with marginal increase in communication overhead.

Some other key observations include:

1. In case of greyhole attacks [7], CTN_RF achieves 100 % malicious node detection in lesser time than the other two strategies. CTN_MI and CTN_RF exhibit stable delivery probability (80–82 %) under all values of malicious probability.
2. Investment in enabling connectivity among trusted nodes can have significant performance benefits both in terms of reduction of communication overhead and fast detection.
3. In terms of false positives and detection time, the proposed strategies perform significantly better than existing strategies FBIDM & MUTON.

All simulations have been performed using enhanced ONE [5] simulator for two different mobility models: random way point movement model (RWP) [1] and shortest path map-based movement model (SPMBM) [9]. We used Spray & Focus [22] as our representative protocol as it scores over PRoPHET [12] in a general network scenario where the traffic does not show extensively periodic behavior. However, with minor modifications, the approach is applicable to various other metric-based algorithms in general.

The rest of the paper has been divided into 9 sections. Section 2 presents a literature survey, followed by Sect. 1, where we model the DoS threat in Spray & Focus routing protocol. Section 4 outlines the mechanism to detect the modeled DoS attack. Section 5 presents the algorithms and the three proposed combat strategies. Extensive simulation based performance evaluation is detailed in Sects. 6, 7 and 8. In Sect. 9, we state our observations regarding the 3 strategies. Finally, the paper is concluded in Sect. 10.

2 Survey of related work

In recent years, there has been significant research in modeling various threats for DTN routing algorithms. Such threats include Intrusion [3], Denial-of-Service (DoS) [17], Greyhole [7], Tailgating [7], Blackhole attack [20], Insider attacks [10], Misrouting Attacks [17] [20], TTL Alternation, Layer injection attack [15, 30], Layer removal attack [15, 30], Submission refusal [30], Free riding attack [15]. Significant efforts have also been made in mitigating these threats by designing secured routing algorithms such as FBIDM [3], MUTON [17], Security in VDTN [7], Encounter Tickets[7], SRSnF [20], Combating Wormhole Attack in DTN [18], Packet Exchange [10], SMART [30], Pi-incentive [15], etc.

As a contrast, marginal work has been carried out in addressing the greyhole [7] attack. Although [7] uses an incentivized approach for mitigating (proactive) this attack, no work has still been done using a reactive approach, to the best of our knowledge. In this paper, we mainly deal with Blackhole and Greyhole Denial-of-Service (DoS) [3, 17] attacks, which translate to an attempt to make a machine, network resource or service unavailable to its legitimate users. There are two general forms: (1) those which crash services, and (2) those which flood services. Our literature survey shows that reactive approaches have so far been designed to combat DoS attacks in most cases (as evident from Table 1).

Chuah et. al. [3] proposed a ferry based detection scheme (FBIDM). In this scheme, detection of malicious nodes involved in DoS attack was performed for the Prophet [12] routing scheme. Detection was carried out by some trusted checkers, called ferry nodes. However, FBIDM did not consider the use of the property of transitivity, which has been shown to be an important attribute for calculating delivery probability. This strategy was improved upon by Ren et. al. [17] by considering the property of transitivity in their strategy called MUTON. MUTON also used ferries to detect malicious nodes and was shown to perform better in terms of detection efficiency when compared to FBIDM. However, both FBIDM and MUTON suffered from large detection overhead.

Meanwhile, Li et. al. [7] proposed a strategy with encounter tickets to thwart blackhole attacks for metric-based routing algorithms. The scheme required forwarding nodes to generate signed encounter tickets to build an encounter history, which was later interpreted appropriately to make forwarding decisions. Ren et. al. [10] proposed a packet exchange recording scheme for ProPHET and MaxProp [2] routing, which required each node to maintain two tables for storing records of packet exchange. Each node was taken through a sanity check at every encounter, failing which the node was blacklisted. Ren et. al. [18] proposed a detection mechanism for Wormhole attacks for ProPHET, which exploited the existence of a forbidden topology structure in the network by utilizing the high mobility of the nodes as such structures cannot exist in legal networks. This approach works effectively in sparse networks and does not need any additional devices.

A strategy using trusted nodes based on table-exchange called SRSnF [20] was proposed by us to combat the Black-Hole DoS attack in DTNs for the Spray & Focus routing protocol. All regular nodes were required to participate in detection. They performed integrity checks on the other nodes of the network. To enable detection, nodes required to exchange all tables maintained by it, thereby incurring huge overhead. Due to the large communication overhead of exchanging tables, nodes often missed contact opportunities with other nodes.

In general, all the reactive strategies attempt to reduce detection time at the cost of increased overhead. In a novel way to reduce overhead, here we propose design of secured strategies which exploit connectivity among trusted nodes. As we consider DoS attack in Spray & Focus routing protocol, in the following section, we provide a brief outline of the protocol and explain how DoS is modeled in it.

3 Denial-of-Service in Spray & Focus routing

In this section, we describe the functioning of the Spray and Focus routing protocol in brief and model the Denial-of-Service threat on it. In DTN, routing consists of a sequence of independent, local forwarding decisions, based on the current connectivity information and predictions of future

Table 1 A comparative summary of strategies based on reactive approaches to combat threats in DTN routing

Scheme	Routing Protocol	Threats Addressed	Limitations
Reactive Mitigation Approach—Mitigate effects of malicious activities			
FBIDM [3]	PRoPHET [12]	DoS, Intrusion	False positives Transitivity not considered
MUTON [17]	PRoPHET [12]	DoS, Intrusion	Specialized nodes required
Encounter Tickets [7]	Metric-based [2, 12, 22]	DoS, Tailgating	False ticket through collusion
SRSnF [20]	Spray & Focus [22]	Blackhole DoS	High overhead involved
Wormhole Attack_DTN [18]	ProPHET [12]	Dos, Insider attacks	Special type of hardware is required
Packet Exchange [10]	ProPHET [12], MaxProp [2]	Insider attacks	High overhead involved

connectivity information. In Spray & Focus [22] routing protocol, the total number of message copies generated per message is bounded without compromising with performance. Whenever any two nodes encounter each other, they exchange their summary vectors and check which packets they have in common. The message forwarding works in two phases *Spray* and *Focus*. A node that owns a forwarding token has the authority to create & forward an extra message copy depending on which phase the routing is in.

Spray phase A node having a message copy along with $n > 1$ corresponding forwarding tokens will work in spray phase. When encountering a node not having any copy of the given message, it forwards a message copy to it. Besides, it transfers $\lfloor n/2 \rfloor$ forwarding tokens to it and keeps $\lceil n/2 \rceil$ for itself.

Focus phase A node having a message copy with only one forwarding token works in the focus phase. The copy is then routed to an encountering node not having a message copy, according to the single copy utility-based scheme with transitivity.

A set of timers are maintained in both i and j which record the time since two nodes i and j last encountered. At every clock tick, each timer is increased by 1. Position information regarding different nodes gets indirectly logged in the last encounter timers, and a lower timer indicates that the destination is expected to be somewhere nearby.

Let the utility declared by node i for node j be denoted as $\tau_i(j)$. Initially, we set $\tau_i(i) = 0$ and $\tau_i(j) = \infty$, for all i, j. Whenever i encounters j, we set $\tau_i(j) = \tau_j(i) = 0$. The utility function of node i for node j can be updated not only by direct encounter, but also by transitivity through another node k, according to the following equation:

$$\tau_i(j) = t_m(d_{ik}) + \tau_k(j) \tag{1}$$

where $t_m(d_{ik})$ is the time required for node i to reach node k under mobility model m. It may be noted that forwarding decisions are based on utility values only; the lower the utility value the better the choice for next hop.

Spray & Focus as implemented in ONE simulator [24] maintains a list of tuples $\langle Message, Connection \rangle$. The list gets updated when events occur. Four events cause the list to be updated: (1) a new message is forwarded, (2) a new message is received, (3) a connection comes up, or (4) a connection goes down. On generation or receipt of a new message by this host, the collection of open connections is examined to see if the message should be forwarded along them. If so, a new tuple is added to the list. Apart from this list, Spray & Focus does not explicitly maintain any table. The nodes in focus phase determine which node they should transfer the message to, by calculating the utility value using above tuples.

The threat model is presented in the next subsection, based on the above fact.

3.1 Threat model

In this paper, we shall consider the Black Hole Denial-of-Service (DoS) attack, i.e., malicious nodes will drop all packets they receive. Each such node will participate in routing of the message but is not going to forward the packet any further.

In the spray phase of the Spray & Focus routing protocol [22], maliciousness can be shown by accepting packets from neighbouring nodes, then dropping them immediately. Let such behaviour be known as *Spray Maliciousness* and nodes exhibiting it be known as *spray malicious nodes*. In focus phase, the forwarding decision is based on utility values. A node can pose as a favourable relay node for other destinations if it can declare lower utility values of itself to those destinations. Hence, in the focus phase, maliciousness may be exhibited by a node by declaring fake (lesser) utility values (less being better) for the destination. Let such behaviour be known as *Focus Maliciousness* and nodes behaving in such fashion be known as *focus malicious nodes*. If successful, such nodes will receive a large fraction of packets from other regular nodes and are going to drop them.

Here, we also consider the greyhole DoS attack, where with certain probability some nodes drop messages and alter routing tables. If the mischievous node's drop probability is 1, it turns into the *Blackhole attack*. If the node's drop/alter table probability is in between 0 and 1, both ends exclusive, then it is called a *Greyhole attack*.

The next section describes the mechanism to detect DoS threats in the Spray & Focus routing protocol by relying on trusted nodes. Specifically, we explain how the lists/tables in the original Spray & Focus routing are modified, new tables added, exchange of information upon an encounter between trusted and normal nodes takes place, and finally detection is achieved.

4 Detection mechanism using trusted node

As mentioned previously, the strategy of checking each node by its peer node during an encounter can incur high overhead. There is a chance that the network may spend more time checking for sanity than doing productive work. When the network is free from malicious activity, or the impact of malicious activity is within tolerable limits, it is highly inappropriate to continue with such a resource-intensive strategy; it may hamper the normal functioning of the network. In order to deal with this problem, and ensuring detection and elimination of malicious nodes at the same time, we use an approach which is based on the utilization of specialized trusted nodes. Such an approach does not burden the regular (normal non-malicious) nodes

with extra computation, but entrusts a certain section of specialized nodes to perform such complicated activities. In the following subsections, the roles of regular and trusted nodes are highlighted, and the detection process is also illustrated with an example.

4.1 Role of a regular node

This subsection discusses about (1) the tables and lists maintained by any regular node to implement secured spray & focus routing and (2) the actions taken by such nodes.

Information maintained by regular nodes Each trusted and normal node maintains a table with entries corresponding to each encounter in the network; this is called ***Network Encounter Table*** **(NET)**. A typical record in a node's NET will have the following fields: (a) Sender node's ID; (b) Receiver node's ID; (c) Time of encounter; (d) Message ID for the message exchanged; (e) Copies remaining with sender after message exchange; (f) Copies remaining with receiver after message exchange. Additionally, each node stores a ***Malice Identification List*** **(MIL)** with node IDs of all nodes in the network which have been detected as malicious. None of these data structures are present in the original Spray & Focus [22]. We also introduce the concept of a ***sink node*** to avoid false detections. Whenever a node is forced to purge a message due to the expiry of its time-to-live (TTL) or due to buffer constraints of the node, it generates a record that states that the receiver is the sink node. Hence, its presence in a record confirms that the message had been removed from the buffer either due to acceptable reasons like TTL expiry or buffer constraints, and not due to a result of malicious intentions.

Action taken by regular nodes Whenever two nodes (say A & X) encounter each other, they perform the following actions [refer Fig. 1(a)], in the specified order:

1. Node A checks its MIL for node ID of X. Node X performs the same MIL check for A. If any of the two tests positive, no further interaction takes place. If both test negative, they move on to the next step.
2. The nodes A and X exchange packets as defined by the Spray & Focus routing protocol.
3. Corresponding records are generated, and these records are inserted into the NETs of A and X.
4. Nodes A and X exchange their MILs with each other. If A finds any node ID in X's MIL that was not already present in A's MIL, then it updates the MIL with those IDs; X does the same for A. Thus, both their MILs get updated with the latest detection status.

The trusted nodes perform some exclusive functions, as described in the following sub-section.

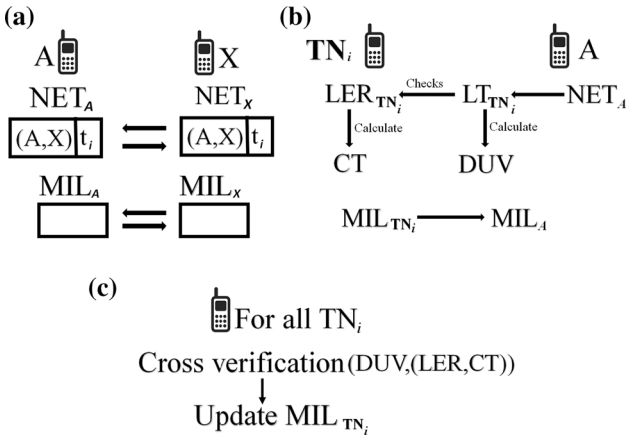


Fig. 1 Schematic view of information exchange in different situations. **a** Encounters between regular nodes, **b** information gathering phase by TN, **c** detection phase by TN

4.2 Role of a trusted node

This subsection discusses about the role of trusted nodes in detecting spray maliciousness and focus maliciousness. The tables and lists maintained by each trusted node, and the corresponding actions taken by such a node to enable spray and focus maliciousness detection, is presented below.

Spray maliciousness detection A Trusted Node (TN) looks for any missing message copies which are unaccounted for as a part of spray phase detection. Whenever TN encounters with a regular node in the network, it receives the corresponding NET from the node. The TN then looks up this NET to find any message copy/copies that has/have been received by the interacting node, but not accounted for (i.e., neither transmitted nor purged). In case such a message copy is found, the corresponding node is tagged as malicious. TN also maintains a Malice Identification List (MIL) which it updates with the node's ID in case there is a successful detection. Otherwise, TN updates the MIL of the interacting regular node with newly added node IDs in its MIL.

Focus maliciousness detection The TN works on the principle of suspicion during the focus phase, i.e., no encounter record is to be trusted unless confirmed by both parties. To this effect, a TN stores encounter records temporarily in a ***Log Table*** **(LT)**. When an encounter record is confirmed by both parties (i.e., interacting nodes), the record is pushed into the ***Latest Encounter Record*** **(LER)** table, and the record is now eligible to be considered for detection computations. On the basis of these confirmed records, the TN computes and stores records of change in utility values by transitivity between nodes in its ***Change by Transitivity*** **(CT)** table. In this case, TN also calculates the utility values based on encounter information which is stored in LT and these particular utility values calculated are stored in ***Declared Utility Vector*** (DUV).

Phases in focus maliciousness detection The focus phase detection activity of a trusted node can be divided into two phases. During the first phase, the TN is busy interacting with participating nodes randomly, and populating its LT, LER, and CT tables on the basis of information accumulated from these nodes. This mode of the TN can be called the ***information gathering mode*** [refer Fig. 1(b)].

Once the TN has sufficient data in its tables, it comes out of its information collection mode and concentrates on using the collected data to compute discrepancies within the network. This mode of the TN can be called the ***detection mode***. During the detection mode, the TN stops interacting with any nodes which may come within range, and focuses entirely on computations [refer Fig. 1(c)]. Both the modes of the TN are alternately activated, i.e., when the TN gathers information, it does so for a fixed period of time, and then changes to the detection mode, which in turn, is over within a period of time, and information gathering mode is reactivated. It is important to note that each time the TN starts with its information gathering mode, it does so with fresh tables, i.e., there is no residual data in its tables. This is to ensure that there are no redundancies in the checking activities.

The next subsection illustrates the detection mechanism with an example.

4.3 Illustration

The spray and focus malicious detection operations have been illustrated separately in the following examples.

Spray maliciousness detection Suppose B is a malicious node, whereas A and C are non-malicious. B drops any message copy that it receives, immediately. Let us consider the following events labelled as E_i:

1. E_0: Node A generates 5 copies of a message with ID M-702 at time 10 units.
2. E_1: Node A encounters node B and transfers 2 tokens for M-702 to it at time 15 units; B being malicious drops them immediately.
3. E_2: Node A encounters node C and transfers 1 token for M-702 to it at time 20 units.
4. E_3: Time-to-live (TTL) for M-702 expires; A, B, and C purge their copies of M-702 at time 30 units.
5. E_4: Node C generates 3 copies of a message with ID M-893 at time 35 units.
6. E_5: Node C encounters with node B and transfers 1 token for M-893 to it at time 50 units; B being malicious drops it immediately.

During E_5, A, B, and C update their NETs as in Tables 2, 3 and 4 respectively. Let us consider following possible cases after E_5:

1. *Case 1* Trusted node encounters node A. From A's summary vector, TN finds that A has no message copies. From A's NET, TN observes that it only had M-702 with it, and all copies of M-702 have been accounted for. Hence, A is a non-malicious node.
2. *Case 2* Trusted node encounters node B. From B's summary vector, TN finds that B has no message copies. From B's NET, TN observes that it had received 2 tokens for M-702 from A, and 1 token for M-893 from C, all of which are unaccounted for. TN tags it as a malicious node.
3. *Case 3* Trusted node encounters node C. From C's summary vector, TN finds that C has 2 tokens for M-893. From C's NET, TN observes that it once had tokens for M-702 with it, but they have all been accounted for. It currently holds 2 tokens for M-893, which matches with its NET records. Hence, C is a non-malicious node.

Focus maliciousness detection Note that we have assumed that there are 4 nodes in the network, with node IDs A, B, C, D. The time line of encounters has been shown in Fig. 2. The Log Table (LT) has been shown in Table 5. It stores unconfirmed encounter records. A record is confirmed only after the trusted node has encountered both parties that claim to have taken part in an encounter, and both of them agree on the Time of Latest Encounter. Each time an encounter record is confirmed, the record is inserted into the LER and it is removed from the LT. The "Confirmation" column represents whether a match has been found and the record has been inserted into the LER; "Yes" indicates confirmation, while "No" indicates that the record is yet to be confirmed. The "Confirmation" column does not exist in the actual implementation; it has been added only for the reader's understanding.

The Latest Encounter Record (LER) table has been shown in Table 6 and the Change by Transitivity (CT) table has been shown in Table 7. We have assumed that $t_m(d_{AB}) = 0.5$. Let us call this factor δ and use this notation hereafter. Note that the resulting matrix is of the order NxN, where N is the number of nodes catered to by the Trusted node (in this example, $N = 4$). Each cell in the matrix contains 2 values:

1. Time of change by transitivity T.
2. $\tau_X(Y)$ value at the time of this change, where X is the node indicated by the row and Y is the node indicated by the column.

It is to be noted that all changes in the CT table are done after a record has been pushed into the LER, post confirmation. For the given time line, the CT table is populated as follows. At time 0 units, B and C encounter, and there is no change by transitivity. At time 10 units, C and D encounter; as a result $\tau_D(B)$ changes as $\tau_D(B) =$

Table 2 Network encounter table with node A after E_5 in spray maliciousness detection

Sender	Receiver	Time of encounter	Message ID	Copies with sender	Copies with receiver
A	B	15	M-702	3	2
A	C	20	M-702	2	1
A	Sink-node	30	M-702	0	2

Table 3 Network encounter table with node B (malicious) after E_5 in spray maliciousness detection

Sender	Receiver	Time of encounter	Message ID	Copies with sender	Copies with receiver
A	B	15	M-702	3	2
C	B	50	M-893	2	1

Table 4 Network encounter table with node C after E_5 in spray maliciousness detection

Sender	Receiver	Time of encounter	Message ID	Copies with sender	Copies with receiver
A	C	20	M-702	2	1
C	Sink-node	30	M-702	0	1
C	B	50	M-893	2	1

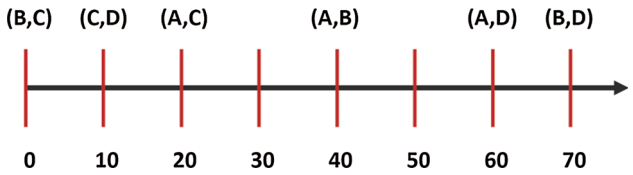


Fig. 2 Event Timeline used to illustrate focus maliciousness detection

Table 5 Trusted node—Log Table (LT) in focus maliciousness detection example

Interacting node pair	Time of latest encounter	Confirmation
A, B	40	Yes
A, C	20	Yes
A, D	60	Yes
B, C	0	Yes
B, D	70	Yes
C, D	10	Yes
A, C	95	No
B, D	115	No

Table 6 Trusted node—Latest Encounter Record (LER) table in focus maliciousness detection example

Interacting node pair	Time of latest encounter
A, B	40
A, C	20
A, D	60
B, C	0
B, D	70
C, D	10

Table 7 Trusted node—Change by Transitivity (CT) Table in focus maliciousness detection example

	A	B	C	D
A	Invalid	(20, 20.5)	–	(20, 10.5)
B	(70, 10.5)	Invalid	(40, 20.5)	(40, 11)
C	–	–	Invalid	-
D	–	(60, 20.5)	(60, 40.5)	Invalid

$\delta(d_{DC}) + \tau_C(B) = (0.5 + 10) = 10.5$, and is entered into CT as (10, 10.5). At time 20 units, A and C encounter; as a result $\tau_A(B)$ changes as $\tau_A(B) = \delta(d_{AC}) + \tau_C(B) = (0.5 + 20) = 20.5$, whereas $\tau_A(D)$ changes as $\tau_A(D) = \delta(d_{AC}) + \tau_C(D) = (0.5 + 10) = 10.5$. The records (20, 20.5) and (20, 10.5) are inserted into CT. Following a similar procedure, the records (40, 20.5), (40, 11), (60, 20.5), (60, 40.5), (70, 10.5) are generated for $\tau_B(C), \tau_B(D), \tau_D(B), \tau_D(C), \tau_B(A)$ respectively.

Let us consider the LT, LER, and CT tables above. Note that $\tau_{declared}(P, Q)$ indicates the utility value declared by node P for node Q during the Focus phase, whereas $\tau_{calculated}(P, Q)$ indicates the utility value of node P for node Q as computed by the Trusted node. The following cases may arise:

1. *Case 1* The trusted node encounters C next. C gives the time of encounter with A as 90. The trusted node finds discrepancy since log says 95. Since a malicious node will try to decrease its utility value, 95 is a more appropriate value to perform maliciousness. Thus, The trusted node concludes that 90 is the correct value and since A sent 95, it is tagged as malicious. A record

with (A,C,90) is inserted in LER and corresponding CT computations are done.

2. *Case 2* The trusted node is in detection mode at time 90. The trusted node is checking sanity for node A, and is currently considering node B as the peer of node A. The trusted node can choose between the record present for the node pair (A,B) in LER and the one present in CT. Since the time of update in case of the LER is 40, and that in case of CT is 20, the former is considered, it being the more recent record. Thus, the trusted node calculates $\tau_{calculated}(A,B)$ as $(90-40)=50$. Suppose $\tau_{declared}(A,B)$ is 45, i.e., A has decided to reduce its utility value for B by 5 units as an exhibition of its malicious behaviour. The trusted node detects anomaly and immediately tags A as malicious.
3. *Case 3* The trusted node is in detection mode at time 90. The trusted node is checking sanity for node D, and is currently considering node C as the peer of node D. The trusted node can choose between the record present for the node pair (D,C) in LER and the one present in CT. Since the time of update in case of the LER is 10, and that in case of CT is 60, the latter is considered, it being the more recent record. Thus, The trusted node calculates $\tau_{calculated}(D,C)$ as $(90-60)+40.5=70.5$. Suppose $\tau_{declared}(D,C)$ is 60, i.e., D has decided to reduce its utility value for C by 10.5 units as an exhibition of its malicious behaviour. The trusted node detects anomaly and tags D as malicious.

The next section presents the algorithms designed based on the detection mechanism discussed in this section.

5 Algorithms and combat strategies

Before the network starts functioning, it is essential for each participating node and the trusted node to be aware about all other participating nodes; otherwise, external intrusion would become rampant and the network would suffer from malicious activities by intruders. To ensure this, there is an Authentication Authority (AA) which assigns a unique node ID and a set of unique public and private keys to each participating regular node, and entrusts the specialized task of detection to the Trusted node. Also, AA ensures that each node (including TN) is preloaded with the list of node IDs all participating nodes, as well as their corresponding public keys. We assume that either this list does not change throughout the working of the DTN, or if there are additions or subtractions, the AA is able to change appropriately settings of all nodes.

Whenever a node generates a record for its NET corresponding to an encounter with another node, it is signed and encrypted using its pair of public and private keys. Such an entry can be decrypted by another authenticated node using its set of keys. Securing information using such techniques is a different field altogether and we do not delve into the details of such techniques in this paper; rather we state that by using such a technique, we can ensure that modification of the contents of any entry is rendered impossible, thus securing the network from information forging attacks.

Furthermore, let us assume that AA generates a unique *sink node ID*, which is never assigned to any participating node, but is included in the list loaded into all nodes. Whenever a node is forced to purge a message from its memory due to buffer constraints, it generates an entry with this *sink node ID* as the receiver node. During detection, the Trusted Node counts all such entries and includes this figure in its evaluation to ensure that there is no false report of malicious activity resulting from purging due to buffer constraints rather than due to malicious intentions.

The following subsections present the detailed detection algorithms.

5.1 Spray phase detection algorithm

The Spray phase detection algorithm, as given in Algorithm 1, is largely self-explanatory. Trusted node first finds out all the unique message IDs that the currently encountered node has held over time, by looping through the *Message ID* column in its NET. Once done, it loops through all unique message IDs, and for each of those, checks whether all tokens for it have been accounted for. If so, it proceeds to the Focus phase detection algorithms. If not so, it tags the currently encountered node as malicious.

5.2 Focus phase detection algorithm–*Information Gathering Mode*

Algorithm 1: Spray Phase Detection Algorithm

```
/*Key: TN has encountered with node P, and obtains NET from it */
for (Every unique message ID that is present in P's NET) do
    if (no.of copies received - no.of copies delivered - no.of entries to sink node) <> 0) then
        Tag P as malicious, and add to MIL;
    end
end
else
    Follow focus maliciousness detection mechanism;
end
```

Algorithm 2: Focus Phase (*Information Gathering Mode*)

```
// TN has encountered P, obtains encounter records from its NET
// DeclaredUtilityVector: vector of vectors for declared util. values
for (Every encounter record obtained from NET of P) do
    X := Node encountered by P;
    T := Time of encounter between P and X;
    if (any record for node pair (X,P) is present in LT) then
        Obtain the Log-Table-Time (LT_time) value for (X,P);
        if (LT_time(X,P) = T) then
            Remove record LT(X,P,T) and insert into LER record (X,P,T);
        end
        else
            // Instant detection for mismatch in LT values
            if (LT_time(X,P) > T ) then
                Remove record LT(X,P,T) and insert into LER record (X,P,T);
                Tag X as malicious;
            end
            else if (LT_time(X,P) < T) then
                Remove record LT(X,P,T) and insert into LER record (X,P,LT_time(X,P));
                Tag P as malicious;
            end
        end
        Obtain LER value for the node pair (P,X);
        if (LER_time(X,P) > T) then
            Continue, since TN already has latest value;
        end
        else
            Update LER(X,Y) with T;
            Calculate transitivity changes, if any, due to the current encounter, as per equation 1;
            Update CT with these calculated changes;
        end
    end
    else
        Insert record for node pair (X,P) in LT;
    end
end
Obtain τ_declared(P, X) for every node X other than P from P's table storing utility values for other nodes;
Store all these values in DeclaredUtilityVector[P];
Add the current time T* to this vector;
```

Algorithm 2 describes the algorithm that the TN executes when it is in the information gathering mode. Suppose the TN has encountered with a node symbolized by P. It obtains the encounter records from the NET of P at the very beginning. Now TN loops through every encounter of P as found in the encounter records, and performs a set of tasks in each iteration.

Algorithm 3: Focus Phase (*Detection Mode*)

```
/* LER_time(X,Y) is the time of latest encounter between X and Y according to LER */
/* CT_time(X,Y) is the time of change by transitivity according to CT */
/* CT_value(X,Y) is the τ_X(Y) value at CT_time(X,Y) according to CT */
for (Every node X in the network) do
    for (Every node Y other than node X) do
        Trusted node (TN) sets τ_declared(X,Y) = DeclaredUtilityVector[X][Y] + {CurrentTime − T*(X)};
        if (LER_time(X,Y) >= CT_time(X,Y)) then
            TN sets τ_calculated(X,Y) = CurrentTime − LER_time(X,Y);
        end
        else
            TN sets τ_calculated(X,Y) = CT_value(X,Y) + {CurrentTime − CT_time(X,Y)};
        end
        if (τ_declared(X,Y) <> τ_calculated(X,Y)) then
            Tag X as malicious;
        end
        else
            Abort and continue;
        end
    end
end
```

Suppose that the node that P had encountered, as found in one of the iterations, is represented as X, and let the time of encounter between P and X (as found in EH obtained from P) be T. Now, the TN searches in the Log Table (LT) for an unconfirmed encounter entry for the nodes P and X.

If such a record is found, then TN has to ascertain whether this newly obtained record from EH matches with the record found from LT. To this end, it obtains the Log-Table-Time (LT_{time}) for the LT record of P and X. If this LT_{time} value for the node pair (P,X) matches with the time T noted from the EH record, then the record is genuine and has been confirmed. The TN immediately deletes the unconfirmed record LT(X,P,T) from the Log Table (LT) and inserts a record with the same values of attributes into

the Latest Encounter Record (LER) table, thereby confirming it.

However, if LT_{time}(X,P) does not match with the noted time T, then the TN assumes that one of the 2 nodes—X or P—is lying, i.e., giving out wrong information. Since a node would give out wrong information only for selfish reasons, we assume that the node which shows a more recent encounter than the other, is the one which is behaving maliciously, since it can get a distinct advantage in utility value by doing so.

TN now determines which one of the 2 nodes has declared a more recent encounter time. If it is P, i.e., the value LT(P,X) is more recent than the time T, then TN assumes that T is correct, and inserts a record into the LER with values (X,P,T). The unconfirmed record LT(X,P,T) is removed. X is then tagged as malicious. If on the other hand, it is X, i.e., the value T is more recent, then TN assumes that the time of encounter LT(P,X) is correct, and inserts a record into LER with values (X,P,LT(P,X)). The unconfirmed record LT(X,P,T) is removed. P is then tagged as a malicious node. This phase can be called the *instant detection phase* since TN does not have to be in its detection mode for such a detection.

Otherwise, if no Log Table (LT) record for P and X is present, then TN searches the LER for the record against nodes P and X. The time of encounter LER_{time}(P,X) is obtained from this record.

If LER_{time}(P,X) is more recent than or as recent as T, then TN concludes that it already has the latest information about the node pair (P,X), and moves on to the next step.

However, if T is more recent than LER_{time}(P,X), then TN has to update its tables with this latest information. The record LER(P,X,LER_{time}(P,X)) is now replaced by the record LER(P,X,T). Next, TN calculates the changes due to transitivity because of the aforementioned update in time of encounter. Once it has computed all such changes, it updates its Change by Transitivity (CT) table with the latest values.

Once all the updates are complete, TN enquires P for the utility values it stores for all other nodes in the network. These utility values are the ones that P would declare when another regular node with a message copy tries to decide whether P is a better relay to the destination, during the focus phase. These values (the value for node X is represented by $\tau_{declared}(P,X)$) are stored in a vector of vectors known as the *DeclaredUtilityVector*. Each vector in this vector of vectors, other than containing N slots for declared utility values (N being the number of nodes in the network), also contains a slot for the time when the current vector had been updated last. This slot is set with the current clock time T^* for the vector corresponding to node P.

The information gathering mode is now complete. TN closes the connection and moves on to find another regular node to perform the same operations on.

5.3 Focus phase detection algorithm—*Detection mode*

During the detection mode, as shown in Algorithm 3, TN makes extensive use of its LER and CT tables. It iterates through the entire set of nodes, generating all possible combinations of node-pairs. The node-pairs were arranged in a matrix where an element i,j corresponds to the comparison of node i with node j. Since comparison of a node to itself is absurd, when the row value is equal to the column value, the TN skips the set of instructions and moves on to the next iteration.

It is also important to note that the iterations are in a row-major fashion, such that, if the first node down a column is A, then all the combinations of node pairs with A as the first node are evaluated first (except the absurd (A,A) pair). Only when A has been evaluated against all the N (total number of nodes in the network) nodes, TN moves on to the next node down the column.

Let us assume that one such non-absurd iteration involves the node pair X and Y (X on the row, and Y on the column). TN sets $\tau_{declared}(X,Y)$ as:

$$\tau_{declared}(X,Y) = DUV[X][Y] + \{CurrentTime - T^*(X)\}$$

where DUV is the *DeclaredUtilityVector*, DUV[X][Y] represents the utility value recorded in the vector corresponding to node X for the node Y, $T^*(X)$ is the time of record in the DUV for the vector corresponding to node X. The previous sub-section has already introduced these parameters.

Now, there can be two cases—either the record in LER is the most updated record for node pair (X,Y), or the one in CT is the most updated one. If LER_{time}(X,Y), i.e., the time of latest encounter between X and Y, is more recent than CT_{time}(X,Y), i.e., the time of latest change by transitivity for pair (X,Y), then the calculated utility value is merely the difference of the current time and the time of encounter LER_{time}(X,Y). Thus, $\tau_{calculated}(X,Y)$ is set as:

$$\tau_{calculated}(X,Y) = CurrentTime - LER_{time}(X,Y)$$

Otherwise, if CT_{time}(X,Y) is more recent than LER_{time}(X,Y), TN concludes that the most recent change in utility value has been by transitivity rather than direct contact. So it sets $\tau_{calculated}(X,Y)$ as:

$$\tau_{calculated}(X,Y) = CT_{value}(X,Y) + \{CurrentTime - CT_{time}(X,Y)\}$$

where CT_{value}(X,Y) is the updated utility value due to the latest change by transitivity.

Now, TN compares $\tau_{declared}(X, Y)$ with $\tau_{calculated}(X, Y)$. If there is a mismatch between the two values, TN tags X as a malicious node. If the values match, no action is taken. The iteration ends here and a new iteration begins.

5.4 Proposed combat strategies

Based on the above algorithms, and depending on what, when and how information is exchanged among trusted nodes as well as among trusted and normal nodes, we propose three strategies i.e. TN (Trusted nodes), CTN_MI (Connected Trusted Nodes with sharing of malicious node information) and CTN_RF (Connected Trusted Nodes with sharing of all the tables) to combat DoS attacks in Spray & Focus routing. Specifically, the actions taken when a (a) a NN encounters a NN, (b) trusted node (TN) or a connected trusted node (CTN) encounters a normal node (NN), and (c) a TN/CTN encounters a TN/CTN, have been mentioned in detail.

Strategy 1: TN This strategy considers a small fraction of nodes as trusted nodes with no interaction between trusted nodes apart from direct contact. It is important to note that the number of trusted nodes that might be allowed to function in a particular network depends largely on the network conditions, and the impact of malicious activities on the network. For example, if malicious activity is rampant in the network, the designer might be inclined to have 10 % trusted nodes in the network. On the contrary, if malicious activity is not significant, 2 % might suffice. The choice also depends largely on whether the network can afford to spend too many of its resources on complex detection computations.

- *Normal Node encounters Normal Node (NN)*: Node checks Malicious Information List (MIL) for node ID of encountered Node. If encountered node is malicious then no further interaction takes place. Otherwise they move on to the next step (exchange packets as defined by the Spray & Focus routing protocol) and both their MILs get updated with the latest detection status.
- *Trusted Node(TN) encounters NN*: Normal nodes share NET table to the TN and TN store it temporarily.
- *TN encounters TN*: Exchanges LT, LER and MIL list.

Strategy 2: CTN_MI Here trusted nodes might be allowed to be connected via an information exchange medium to a central data-center so that they exchange malicious nodes' information list.

- *NN encounters NN*: Same as in TN.
- *CTN encounters NN*: Normal nodes share NET table to the CTN and CTN store it temporarily.
- *CTN encounters CTN*: Always connected. Exchanges LT, LER and MIL list when any modification is observed. The CTNs are allowed to be connected via an information exchange medium to a central data-center so that they exchange malicious nodes Information list. Whenever a malicious node is detected, the nodes ID will be stored in a central database(Malicious Information List) which would act like a global list, and every trusted node will be allowed to access this data. This would help the trusted nodes avoid execution of detection algorithms on nodes which have already been tagged as malicious by other trusted nodes.

Strategy 3: CTN_RF Yet another variation in CTN strategy that we introduce goes like this: Instead of sharing the final information about the list of malicious nodes, we start sharing information beforehand, which the trusted nodes can then use to calculate whether a given node is malicious or not.

- *NN encounters NN*: Same as in TN.
- *CTN encounters NN*: Normal nodes share NET table to the CTN and CTN store it temporarily.
- *CTN encounters with CTN*: Always connected. The tables which contain the Latest Encounter Record (LER) (refer. Table 7) along with the Changes by Transitivity table (CT) (refer. Table 8), are implemented using the Graph Data Structure. The nodes act as vertices and the edges store their encounter information. Apart from storing the utility values declared by the nodes, the edges also store the timestamps T (refer Algorithm 2 for Focus Phase information gathering mode) at which the values have been modified. Cross-verification of the declared utility value and the calculated utility value helps the connected trusted checker to identify malicious nodes early and with better accuracy. This checking of malicious activity in real time helps all trusted checkers to get the updated malicious node list (MIL) in real time.

As is common to all the three strategies, TN/CTN periodically updates its local Log Table (LT) and LER table (encounter record is confirmed by both parties) using Information Gathering algorithm (Refer Algorithm 2) and then detects malicious nodes using the detection algorithm (Refer algorithm 3).

6 Performance evaluation

This section presents a comparative analysis of the three variations of the trusted-node based approach presented by us in this paper. We also compare our proposed strategies with the table-based approach in SRSnF [20], the only related strategy available in the literature.

Table 8 Simulation Setup

Parameter	Value	Parameter	Value
Simulation area	4500 m × 3400 m	Packet size	50 kB–1 MB
Message TTL	8 h	Total no. of nodes	150
Buffer size	2000MB	No. of simulations runs for each experiment	20
Simulation time	24 h	Node speed range	0.5–13 m/s
Initial token no.	3	Message load range	500, 1000, 2000, 3000
Interface 1	Bluetooth (10 m, 2 Mbps)	Interface 2	Satellite (512 kbps)
No. of malicious nodes	20 (spray malicious) and 20 (focus malicious)	No. of trusted nodes	10
Mobility model 1	Random Waypoint [3]	Mobility model 2	Shortest path map based [10]

Simulation setup ONE [9, 24] simulator has been used to perform all simulations. Although ONE is equipped with several DTN routing algorithms and mobility models, suitable changes were required in order to enable it to simulate different secured routing strategies. The modifications as done by us can be accessed by downloading the enhanced modules from our project site [5]. The introduction of malicious nodes and trusted nodes has been done in the `DTNHost.java` file with the activities of both being described in the `DecisionEngineRouter.java` file.

Table 8 gives the simulation parameters and their corresponding values. Message load refers to the total number of packets created throughout simulation time of 24 h, at a uniform rate. Message load has been varied to study the impact of load on different performance metrics.

Evaluation metrics The key objective is to analyze and identify the efficiency of malicious node detection in terms of time consumed, delivery probability and overhead. We consider four metrics as performance indicators, namely, (i) number of malicious nodes detected, (ii) time taken to detect, (iii) communication overhead, (iv) delivery probability. The evaluation metrics are studied under varying load conditions for SRSnF and the three proposed strategies TN, CTN_MI and CTN_RF.

6.1 Results and analysis

The four strategies to be compared have been executed under RWP and SPMBM mobility models. It may be noted that in order to maintain a fair ground for comparison, we have modified the previous SRSnF strategy to the extent that instead of all the regular nodes participating in malicious node detection, only a fraction of them get involved. This fraction is pre-decided such that it is same as the ratio between the number of trusted nodes and the number of regular nodes with respect to TN and CTN_MI. The following subsections provide the results of the evaluation metrics.

6.1.1 Number of nodes detected

This metric calculates the total number of malicious nodes (spray malicious and focus malicious combined) that have been successfully detected during the entire simulation time of 24 h. Figures 3(a) and (b) present the number of nodes detected by the four strategies using RWP and SPMBM movement models, respectively, under varying load conditions.

A closer look at both plots reveals the following facts: Firstly, although strategies TN, CTN_MI and CTN_RF perform better than SRSnF, CTN_RF outperforms all others significantly. This implies that exchange of additional tables LT and LER among trusted nodes via satellite interface can have significant impact on performance. Despite sparseness and intermittent contacts, interface 2 helps the trusted nodes to detect any malicious attempt made by the nodes in the entire service area almost instantly so that they can locally perform computation to detect maliciousness as fast as possible. Exchange of MILs is not required here. Secondly, the number of detections made by TN and CTN_MI is almost same. This means that exploiting connectivity via interface 2 to merely exchange MIL among trusted nodes, so as to inform about the detection of a new node, has no significant impact in overall detection. Thirdly, for all strategies, the detection is significantly better in SPMBM mobility model [Fig. 3(b)] and for CTN_RF, all malicious nodes are detected within the simulation time with full certainty (zero standard deviation). This is because SPMBM provides more contact opportunities compared to RWP which enables better detection. Fourthly, the detected node count reduces slightly as message load increases. Duration of an encounter is shared towards actual packet forwarding and exchange of tables; as load increases, many packets miss the opportunity to get forwarded. Finally, the number of nodes detected by CTN_RF is (20–42) and (12–35) % more compared to TN/CTN_MI, under RWP and (54–67) and (17–56) % more compared to TN/CTN_MI under

SPMBM mobility. Hence, CTN_RF outperforms others significantly.

The simulations assume that 20 nodes exhibit spray maliciousness. A thorough analysis of the logs reveals that in RWP, almost 75–85, 88–92, 85–95 and 90–95 % spray malicious nodes are detected, respectively by SRSnF, TN CTN_MI and CTN_RF. Using SPMBM the same increases to 90–100, 90–100, 90–100 and 100 %. Among 20 focus malicious nodes, the logs reveal that in RWP only 5, 10, 10–20 and 75–85 % such nodes are detected, respectively by SRSnF, TN, CTN_MI and CTN_RF, whereas using SPMBM the same increases to 50–70, 60–70, 60–85 and 100 %. This implies detection of focus maliciousness is more challenging as identifying manipulation of utility values is more difficult.

6.1.2 Time to detect maliciousness

Time of detection refers to the amount of time taken to detect a certain number of malicious nodes which includes both node types—spray malicious and focus malicious. Figures 4(a) and 4(b) present plots of time of detection under RWP mobility model, for message loads 500, 1000 and 2000, 3000 respectively. The plots reveal the following facts: Firstly, during the simulation time of 24 h, except CTN_RF which detects nearly 80 % malicious nodes, other strategies fail to detect even 40–65 % nodes. Secondly, in relatively low load, CTN_RF detects a certain number of nodes in almost 20–30 % less time compared to other strategies. Thirdly, detection time taken by TC and CTN_MI are almost same which is slightly lesser than SRSnF. Moreover, the difference is more prominent in higher load conditions. Our analysis further reveals that spray malicious nodes are detected much earlier than the focus malicious ones. Moreover, a very small fraction of focus malicious nodes are detected by SRSnF, TN and CTN_MI; hence the gradient of the plot corresponding to these strategies is mainly attributed to spray maliciousness.

Figures 5(a) and (b) present plots of time of detection under SPMBM mobility model, for message loads 500, 1000 and 2000, 3000 respectively. In contrast to the RWP mobility model, the plots here reveal two distinctly visible slopes which are attributed to the detection of spray malicious (initial phase) and focus malicious (later phase) nodes. The time to detect spray malicious nodes is same for all strategies and is completed within 2–3 h. This means that the relatively predictable movement pattern along paths enables easier detection of spray malicious nodes. However, the focus malicious detection time differs significantly across strategies. CTN_RF detects the focus malicious nodes in no time whereas increasingly steeper slopes are observed for TN, CTN_MI and SRSnF. As all nodes are detected by CTN_RF within 5 h, which is much less than the message TTL (8 h), the strategy can efficiently handle situations where in the system the node pool is changing, i.e., nodes join and leave frequently.

In general, all strategies perform better in SPMBM compared to RWP. This can be attributed to the fact that RWP involves random movement of the nodes, and therefore detections largely depend on whether two nodes (trusted node or otherwise), which can affect a detection, come into contact. Randomness often proves to be a hindrance to uniform detection since trusted nodes may not come in contact with nodes which hold information necessary for detection for a long period of time. SPMBM on the other hand, is based on targeted movement patterns, and therefore results towards more meaningful encounters. This has a positive impact in malicious node detection.

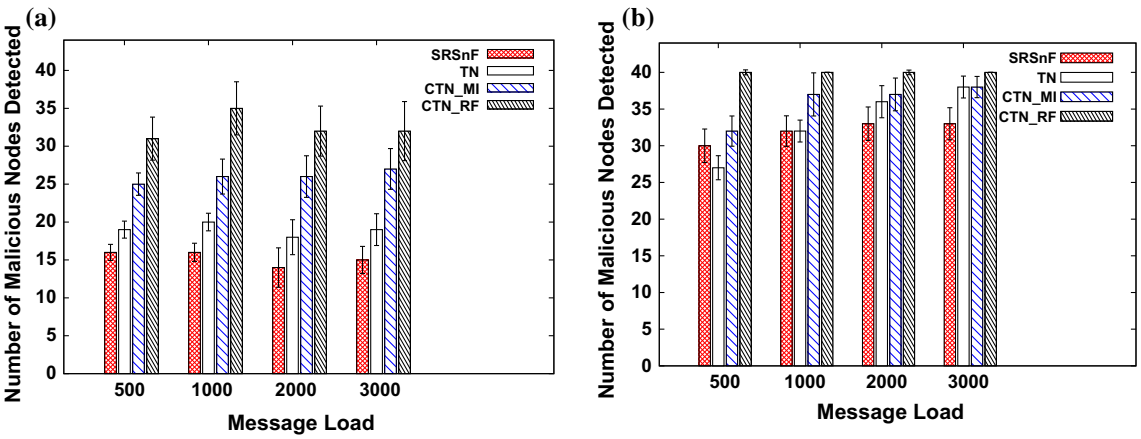


Fig. 3 Plots of detected malicious (spray and focus combined) node count under varying load. **a** Detected malicious node count versus Message load (using RWP mobility model), **b** detected malicious node count versus Message load (using SPMBM mobility model)

6.1.3 Communication overhead

Communication overhead indicates data transfer cost in terms of the tables, lists, etc. exchanged between any two nodes (normal/trusted) upon an encounter, to enable maliciousness detection. It excludes the cost of normal forwarding of message packets due to DTN routing. Figures 6(a) and (b) represent the communication overhead obtained for the TN and CTN_MI strategies under the RWP and SPMBM movement models, respectively.

The following facts can be noted from the figures: Firstly, the overhead in SRSnF is several orders of magnitude higher than that in either of TN, CTN_MI or CTN_RF. This is because of the fact that in TN, CTN_MI and CTN_RF, trusted nodes are specialized dedicated nodes which only participate in detection, whereas a significant fraction of regular nodes perform detection activities in SRSnF, in addition to their regular routing activities. Relatively larger node participation in detection results in the increase of overhead. Moreover, the nodes in SRSnF exchange the entire knowledge base that they maintain/update locally. However, previous plots in Figs. 4 and 6 show this additional overhead fails to provide a better detection performance which indicates that (1) deployment of trusted nodes are beneficial, (2) instead of exhaustive local sharing it is better to invest the overhead in sharing information globally through long range links. This justifies our intuitions behind proposing a connected trusted node based approach. Secondly, although CTN_RF demands more volume of information exchange (through LT, LER and CT tables) compared to TN & CTN_MI, the difference in overhead among TN, CTN_MI and CTN_RF is not significant. This is due to the fact that detection is much faster in CTN_RF, which means that the higher overhead in the initial stages soon gets balanced out in the post-detection phase. The overheads incurred by TN and CTN_MI are comparable as the additional overhead of exchanging malicious nodes list in CTN_MI is negligible. Thirdly, the overhead of the strategies in SPMBM is much lesser than RWP as the nodes meet each other less frequently in RWP due to random mobility.

6.1.4 Delivery probability

Delivery probability at a particular instant refers to the ratio between the number of packets delivered successfully to destination and the number of packets created till that instant. Delivery probability is expected to increase with time as malicious nodes are detected and eventually eliminated in packet routing. The change of delivery probability with time for different strategies has been plotted in Figs. 7(a), (b), using mobility models RWP and SPMBM respectively, considering that 3000 packets are generated during the simulation at a uniform rate. To judge the qualitative gain in performance achieved by the secured routing strategies SRSnF, TN, CTN_MI or CTN_RF, we also show the delivery probability of original Spray & Focus in presence of malicious nodes.

The following observations may be noted: Firstly, using secured routing algorithms, the delivery probability is found to be enhanced by 30 and 50 %, in RWP and SPMBM mobility models respectively. Hence design of suitable combat strategies is capable of curbing the ill effect of maliciousness to a large extent, even while connectivity is intermittent. Secondly, although during the initial stage all strategies produce similar delivery probability, at a later stage, CTN_RF performs slightly better that others. The reason is apparent from the difference in time reuired to detect the focus malicious nodes by CTN_RF with respect to others. Thirdly, the delivery

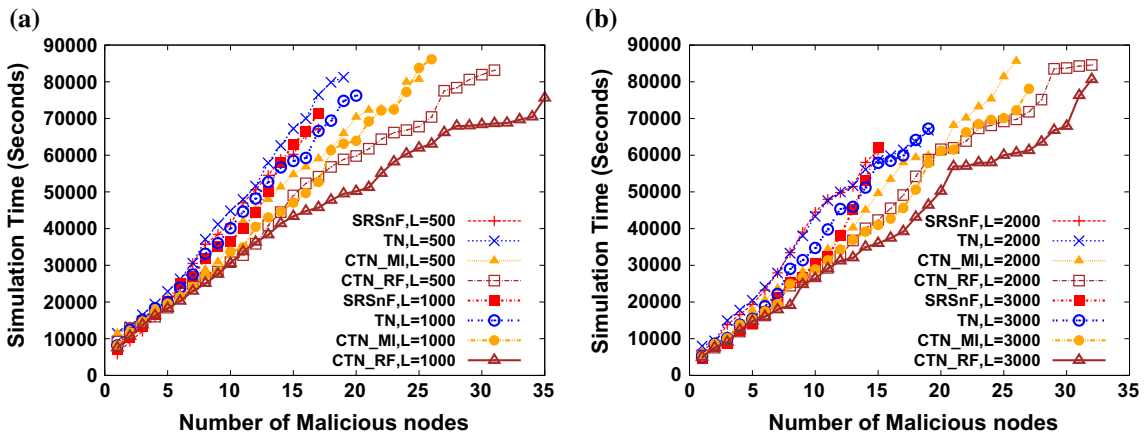


Fig. 4 Plot of Time of detection under RWP mobility model. **a** Detection time versus malicious node count for Message load (L) 500 and 1000, **b** detection time versus malicious node count for Message load (L) 2000 and 3000

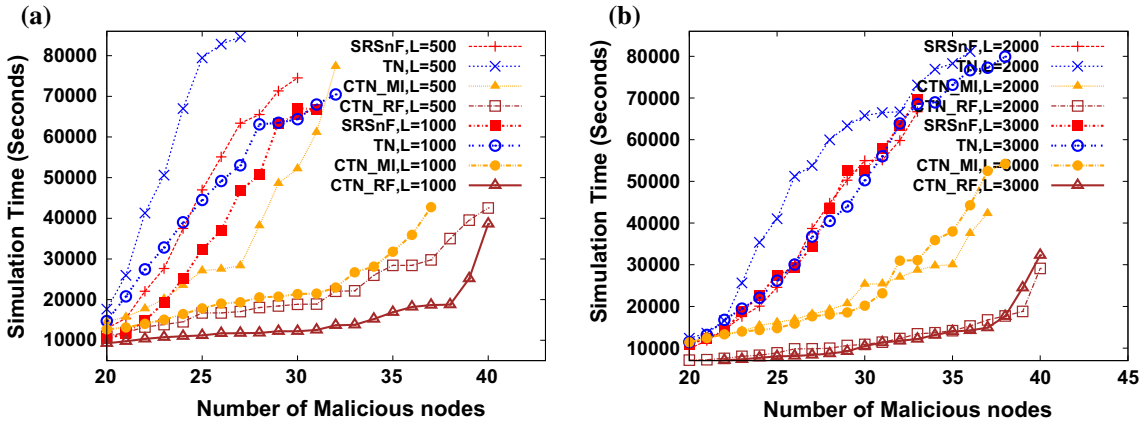


Fig. 5 Plot of time of detection under SPMBM mobility model. **a** Detection time versus malicious node count for Message load (L) 500 and 1000, **b** detection time versus malicious node count for Message load (L) 2000 and 3000

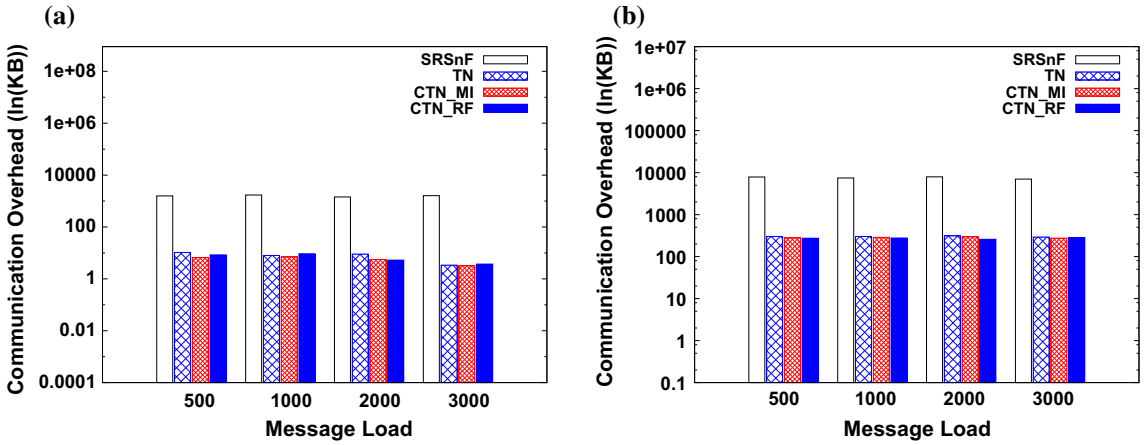


Fig. 6 Plots of communication overhead under varying load conditions. **a** Communication overhead versus load (log-lin scale) under RWP mobility model. **b** Communication overhead versus load (log-lin scale) under SPMBM mobility model

probability in case of SPMBM is much higher than RWP. We have also observed that with load, the mean number of relays (mean number of forwards each packet experiences under varying load) decreases; however the number of relays are always higher in SPMBM compared to RWP which results in better delivery probability.

6.2 Impact of variation of number of trusted nodes

It is observed from Fig. 7(c) that increase in the number of trusted nodes results in reduction of time required to detect malicious nodes. Moreover, the total number of malicious nodes also increases. If we go on increasing the number of trusted nodes, at a certain point the system reaches saturation, where further increase in trusted nodes does not affect the performance significantly. We observed that given a certain area of coverage, the choice of the number of trusted nodes will depend on the density of the normal nodes and malicious nodes.

In the next section, we analyse the greyhole attack in our proposed approaches under varying malicious probability.

7 Impact of greyhole attack on TN, CTN_MI and CTN_RF approaches

In the plots presented in this subsection, all the three strategies that are used for comparison are executed under SPMBM mobility model.

7.1 Delivery probability

Figure 8(a) shows the impact of change in probability of maliciousness on the delivery probability of the network. Here we observe that, for TN approach, little change in delivery probability (varying between 0.81 to 0.83) is observed as malicious probability varies from 0 to 0.3, but it falls considerably as the probability of maliciousness

increases further. However, in the other two strategies (CTN_MI and CTN_RF), delivery probabilities are almost independent of the malicious probability due to the sharing of information among nodes.

7.2 Detection time of malicious nodes

Figure 8(b) shows the impact of variation of malicious probability on detection time. We observe that there is not much impact on the detection time in case of the TN approach when the nodes act as greyhole nodes. But it drops considerably in other two approaches if the nodes are approaching towards the black-hole.

7.3 Percentage of malicious nodes detected

Figure 8(c) shows the impact of different values of malicious probability on the number of malicious nodes detected. We observe that almost all the malicious nodes are detected in our advanced approach (CTN_RF) irrespective of the malicious probability. But for all the other approaches, number of nodes detected varies with variation in node malicious probability.

In the next section, performance comparison is performed between the proposed strategies and some existing strategies, viz. FBIDM & MUTON. However in the following study, we exclude SRSnF as it is a non-ferry based approach.

8 Performance comparison with FBIDM & MUTON in Blackhole attack

In existing literature, both FBIDM & MUTON are analysed using ProPHET routing protocol [12]. In this section, we discuss the original FBIDM & MUTON combat strategies along with the required modifications to be made for use on Spray & Focus routing algorithm. Finally, we present a comparative analysis of the strategies FBIDM, MUTON, TN, CTN_MI, CTN_RF for both RWP and SPMBM mobility models.

FBIDM is a ferry based intrusion detection mechanism for DTNs, over the ProPHET routing algorithm. However, FBIDM does not consider the use of transitivity, which has been shown to be an important attribute in estimating utility values. As in the FBIDM scheme, MUTON also uses the ferry node to perform the intrusion detection function in sparse ad hoc networks but it considers the transitivity property. Also, MUTON is light-weight compared to the FBIDM.

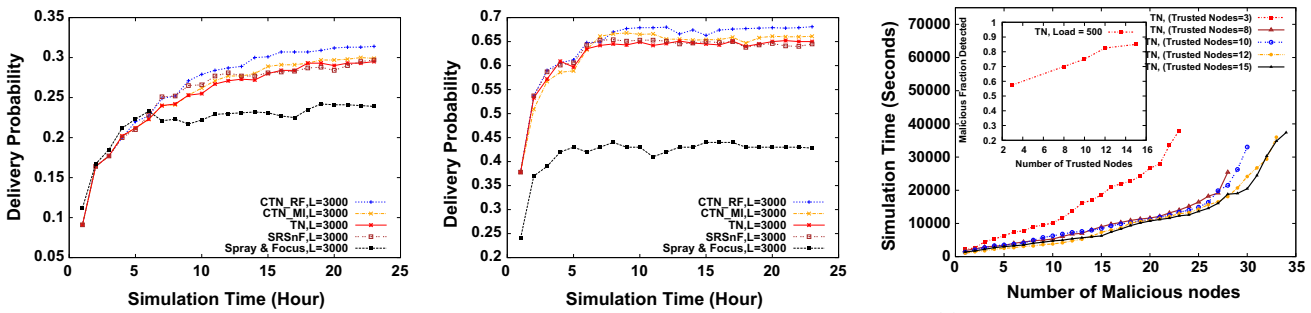
In addition to the four metrics introduced in a previous section, we also measure the fraction of false positive nodes detected, which is the normal nodes detected mistakenly as malicious.

Delivery probability In terms of delivery probability, there is no significant difference in performance for all of the strategies in RWP movement model as shown in Fig. 9(a). In this mobility model, delivery probability is less for all the strategies due to the random movement and less number of encounters. SPMBM provides slightly better result in our strategy as shown in Fig. 9(b).

Count of malicious nodes detected FBIDM performs the worst in RWP movement model as shown in Fig. 10(a). CTN_RF is the best performing strategy. The remaining strategies have almost similar performance in both the movement models as shown in Figs. 12(a)& 10(b).

Our approach is dependent on the LER and transitivity calculations for detection, while FBIDM & MUTON also make use of the values provided by the nodes during the time of encounter. Hence, if the number of encounters is high, as in SPMBM, the number of detections is high for FBIDM and MUTON. However, detections are lesser in our case because not all encounters help to update the LER considerably to result in detection. But as we can see for CTN_RF, when this information is shared among the CTNs, the detection is considerably high.

False positive ratio No false positive nodes are detected in our approach using RWP movement model, but a small



(a) Cumulative Delivery Probability vs. time under message load (L) 3000

(b) Cumulative Delivery Probability vs. time under message load (L) 3000

(c) Impact of Variation of Trusted Nodes

Fig. 7 **a**, **b** Plots of delivery probability over time using SPMBM mobility model, **c** impact of trusted nodes

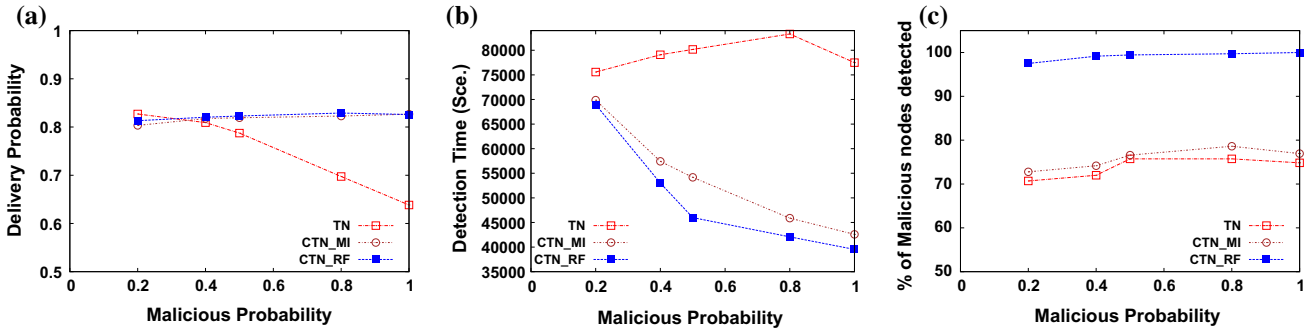


Fig. 8 Different QoS measurement under varying probability in SPMBP movement model in greyhole attack. **a** Delivery probability, **b** detection time, **c** malicious(%) nodes detection

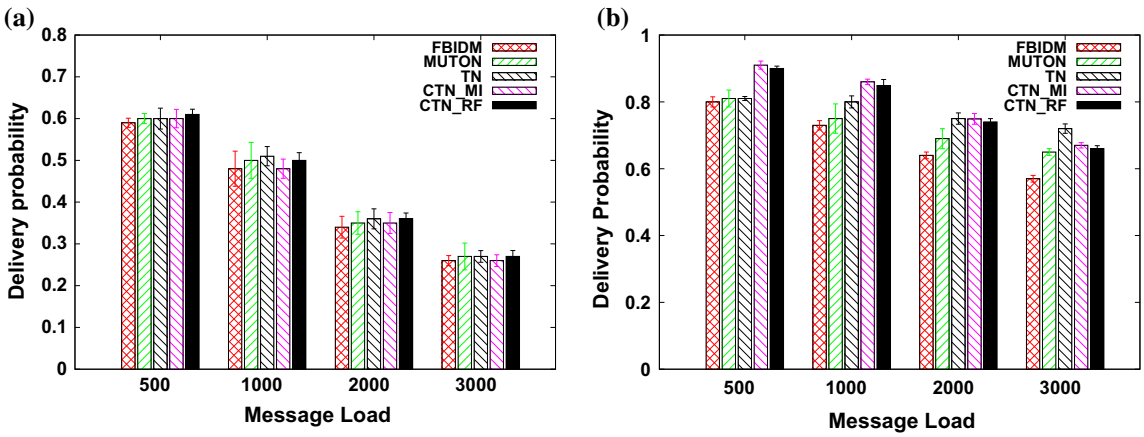


Fig. 9 Plots of Delivery Probability over Load. **a** Delivery probability versus load using RWP mobility model, **b** mean number of relays versus load using SPMBM mobility

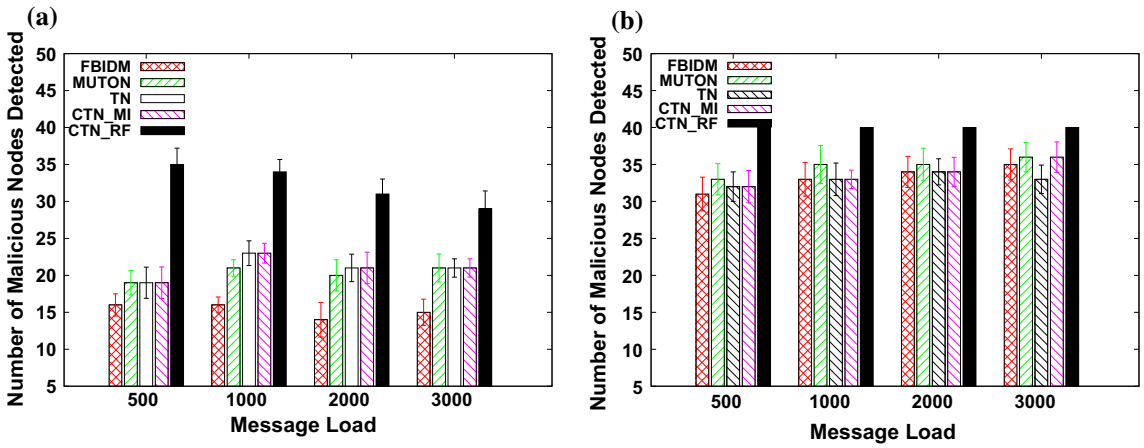


Fig. 10 Plots of detected malicious (spray and focus combined) node count under varying load. **a** Detected malicious node count versus Message load (using RWP mobility model), **b** detected malicious node count versus Message load (using SPMBM mobility model)

fraction of false positives (2–3 %) are still reported in FBIDM & MUTON as shown in Fig. 11(a). However, in the SPMBM model, almost 10–15 % false positives are reported by FBIDM & MUTON as shown in Fig. 11(b), which is significant. The reason is that in FBIDM, the ferry assumes that the most recent $P(i, j)$ is valid which might not always be the case. Hence, this might lead to a false positive scenario. MUTON being developed in a similar

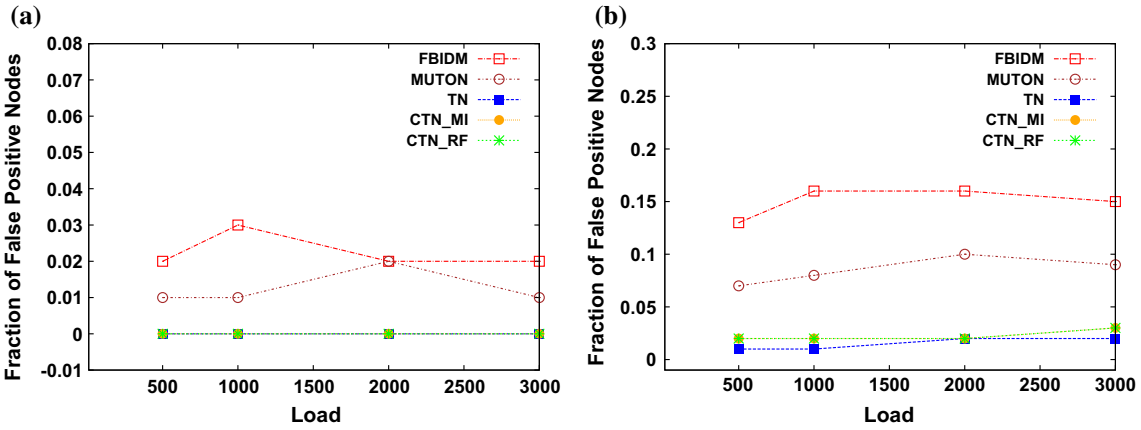


Fig. 11 Plots of false positive ratio under varying load. **a** False positive ratio versus Message load (using RWP mobility model), **b** false positive ratio versus Message load (using SPMBM mobility model)

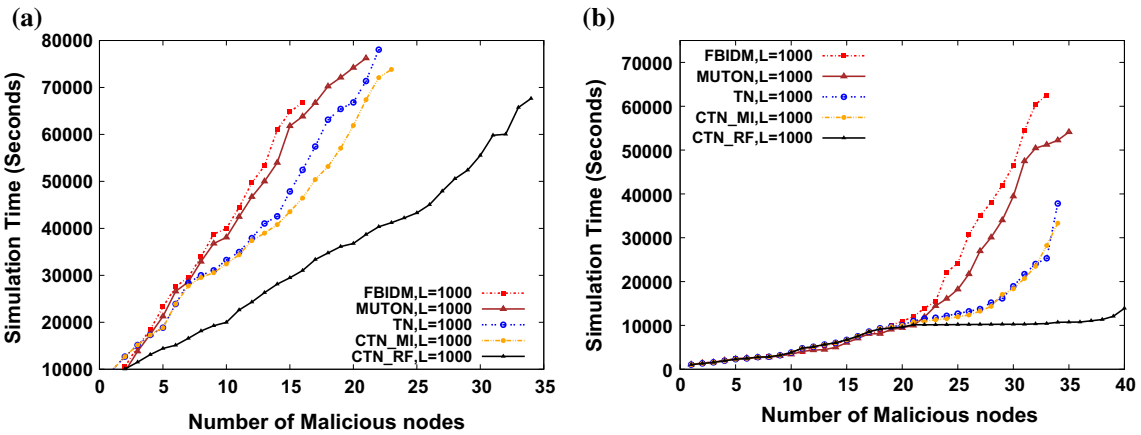


Fig. 12 Plot of detection time versus malicious node count. **a** Detection time versus Malicious node count for Message load (L) 1000 under RWP mobility model, **b** detection time versus malicious node count for Message load (L) 1000 under SPMBM mobility model

context like FBIDM, also results in a handful of false positive scenarios; however, this is lesser because of the use of transitivity value calculation by the ferry. On the other hand, as the proposed strategies are fully dependent on the transitivity values and LER, hence the possibility of false positives is quite less.

Time to detect maliciousness In terms of time to detect a certain fraction of nodes, both FBIDM & MUTON take much more time compared to our proposed strategies. The gain in performance is much better in SPMBM (40 % less time) than RWP (10–15 % less time) mobility model as shown in Figs. 12(a), (b).

To summarize, although in terms of delivery probability and malicious node detection count FBIDM and MUTON exhibit similar performance as our proposed strategies, but in terms of false positives and time to detect, the proposed strategies perform significantly better. It may be noted that with the passage of time, the false positives if not checked will lead to sparseness which in turn may reduce delivery probability. Fast detection will make the system robust in presence of node churn.

9 Observations

The observations drawn from the simulation results can be summarized as follows:

- Under DoS threat, suitably designed combat strategies can improve the delivery probability by upto 50 %.
- Compared to the table-based approach (SRSnF), communication overhead is significantly reduced by employing specialized trusted nodes. In general, the benefits of trusted node based approach is more prominent under shortest path map based mobility

model where nodes meet each more frequently compared to random waypoint model.

- Similar performance of TN and CTN_MI shows that mere exchange of malicious node list information among the trusted nodes does not have additional benefits.
- The capability of node detection at a fast rate (sometimes even less than message TTL) makes CTC_RF a good choice to be deployed in dynamic scenarios where the nodes frequently leave/join the system.
- CTN_RF outperforms CTN_MI by detecting (15–70) % more nodes at a much faster rate (50 % less time) consuming almost the same communication overhead. Hence, exchange of several raw fields among trusted nodes is the key to the design of efficient as well as light-weight combat strategies.
- Our strategies perform in a superior manner, with CTN_RF achieving cent percent malicious node detection in case of blackhole attack and 72–80 % even in case of greyhole attack, in 50 % lesser time as compared to TN.
- In greyhole attack, CTN_MI and CTN_RF provide 80–82 % delivery probability under almost all values of node malicious probability.
- FBIDM & MUTON detect 10–15 % more false positive nodes as compared to our proposed strategies.
- TN detects 10–20 % more malicious nodes in almost same time as compared to FBIDM & MUTON under SPMBM mobility model.

10 Conclusion and future work

In this paper, we have taken a look at the proposed schemes which aim at combating various malicious attacks in a DTN environment. We have revisited a table-based strategy that allows every node to check its peer for maliciousness. We have proposed novel strategies where specialized trusted nodes are given the task of detection, relieving the regular nodes from extra computation. The intricacies of the approach and its variations have been described in detail.

Simulation results show us that the trusted node based approaches are effective in bringing down data overhead considerably. It also scores higher than the table-based approach in terms of detection speed. While there is no significant advantage in terms of achieved delivery probability, there is no drawback either. We further observe that the two variations TN and CTN_MI are more effective than table-based strategy SRSnF in detecting more nodes while incurring less communication overhead. An interesting variation of the trusted node based approach i.e. CTN_RF can be not only to share the detected nodes' information, but also to share the entire detection information that it collects over time. This would mean that every node, no matter which trusted node's area it lies in, will be under the watch of all trusted nodes. Each trusted node will no longer be confined to implementing detection algorithms locally on the nodes in its area; instead it will be allowed to check each and every participating node through information collected jointly. The CTN_RF strategy yields almost all the nodes detected in significantly less amount time and causes near-equal overhead as compared to all other strategies.

While the trusted node based approach is able to curb multiple drawbacks, it is unable to detect groups of malicious nodes colluding together for selfish or malicious reasons. In order to deal with the issue of collusion among malicious nodes, we are also investigating an approach where the trusted node can detect a group of nodes from the connectivity graph rather than detecting a single node at a time.

Other than the nature of attack considered and dealt with in this paper, there are a variety of other attacks possible in a delay tolerant environment. Although a large amount of work has been carried out with regards to security in wireless ad-hoc networks, much remains to be done to combat similar and dissimilar security threats in disruption tolerant networks. Development of strategies to deal with a multitude of other attacks can be fodder for future research as well.

References

1. Bettstetter, C., Resta, G., & Santi, P. (2003). The node distribution of the random waypoint mobility model for wireless ad hoc networks. *Mobile Computing, IEEE Transactions on*, *2*(3), 257–269.
2. Burgess, J., Gallagher, B., Jensen, D., & Levine, B. N. (2006). Maxprop: Routing for vehicle-based disruption-tolerant networks. *INFOCOM*, *6*, 1–11.
3. Chuah, M., Yang, P., & Han, J. (2007). A ferry-based intrusion detection scheme for sparsely connected ad hoc networks. In *Mobile and Ubiquitous Systems: Networking & Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on* (pp. 1–8). IEEE.
4. Dvir, A., & Vasilakos, A. V. (2010). Backpressure-based routing protocol for DTNs. In *Proceedings of the ACM SIGCOMM 2010 Conference, SIGCOMM '10* (pp. 405–406). New York, NY: ACM. doi:10.1145/1851182.1851233.
5. E-one for security. http://www.nitdgp.ac.in/MCN-RG/eONE/security/security.html.
6. Fall, K. (2003). A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 27–34). ACM.
7. Guo, Y., Schildt, S., & Wolf, L. (2013). Detecting blackhole and greyhole attacks in vehicular delay tolerant networks. In

Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on (pp. 1–7). IEEE.
8. Jing, Q., Vasilakos, A. V., Wan, J., Lu, J., & Qiu, D. (2014). Security of the internet of things: Perspectives and challenges. *Wireless Networks*, *20*(8), 2481–2501. doi:10.1007/s11276-014-0761-7.
9. Keränen, A., Ott, J., & Kärkkäinen, T. (2009). The one simulator for dtn protocol evaluation. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques* (p. 55). Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering (ICST).
10. Li, F., Wu, J., & Srinivasan, A. (2009). Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets. In *INFOCOM 2009, IEEE* (pp. 2428–2436). IEEE.
11. Li, Q., Zhu, S., & Cao, G. (2010). Routing in socially selfish delay tolerant networks. In *INFOCOM, 2010 Proceedings IEEE* (pp. 1–9). IEEE.
12. Lindgren, A., Doria, A., & Schelén, O. (2003). Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, *7*(3), 19–20.
13. Liu, B., Bi, J., & Vasilakos, A. V. (2014). Toward incentivizing anti-spoofing deployment. *Information Forensics and Security, IEEE Transactions on*, *9*(3), 436–450.
14. Liu, J., Li, Y., Wang, H., Jin, D., Su, L., Zeng, L., et al. (2016). Leveraging software-defined networking for security policy enforcement. *Information Sciences*, *327*, 288–299. doi:10.1016/j.ins.2015.08.019.
15. Lu, R., Lin, X., Zhu, H., Shen, X. S., & Preiss, B. (2010). Pi: A practical incentive protocol for delay tolerant networks. *Wireless Communications, IEEE Transactions on*, *9*(4), 1483–1493.
16. Ning, T., Yang, Z., Xie, X., & Wu, H. (2011). Incentive-aware data dissemination in delay-tolerant mobile networks. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on* (pp. 539–547). IEEE.
17. Ren, Y., Chuah, M. C., Yang, J., & Chen, Y. (2010). Muton: Detecting malicious nodes in disruption-tolerant networks. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE* (pp. 1–6). IEEE.
18. Ren, Y., Chuah, M. C., Yang, J., & Chen, Y. (2010). Detecting wormhole attacks in delay-tolerant networks [security and privacy in emerging wireless networks]. *Wireless Communications, IEEE*, *17*(5), 36–42.
19. Saha, S., Nandi, S., Paul, P. S., Shah, V. K., Roy, A., & Das, S. K. (2015). Designing delay constrained hybrid ad hoc network infrastructure for post-disaster communication. *Ad Hoc Networks, 25* (Part B), 406–429. doi:10.1016/j.adhoc.2014.08.009. http://www.sciencedirect.com/science/article/pii/S1570870514001802. New Research Challenges in Mobile, Opportunistic and Delay-Tolerant NetworksEnergy-Aware Data Centers: Architecture, Infrastructure, and Communication.
20. Saha, S., Verma, R., Sengupta, S., Mishra, V., & Nandi, S. (2012). SRSnF: a strategy for secured routing in spray and focus routing protocol for DTN. In *Advances in Computing and Information Technology* (pp. 159–169). Springer.
21. Shu, Z., Wan, J., Li, D., Lin, J., Vasilakos, A. V., & Imran, M. (2016). Security in software-defined networking: Threats and countermeasures. *Mobile Networks and Applications*. doi:10.1007/s11036-016-0676-x.
22. Spyropoulos, T., Psounis, K., & Raghavendra, C. S. (2007). Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pp. 79–85. IEEE.
23. Spyropoulos, T., Rais, R. N. B., Turletti, T., Obraczka, K., & Vasilakos, A. (2010). Routing for disruption tolerant networks: Taxonomy and design. *Wireless Networks*, *16*(8), 2349–2370. doi:10.1007/s11276-010-0276-9.
24. The one simulator. http://www.netlab.tkk.fi/tutkimus/dtn/theone/.
25. Vasilakos, A. V., Zhang, Y., & Spyropoulos, T. (2011). *Delay tolerant networks: Protocols and applications* (1st ed.). Boca Raton, FL: CRC Press Inc.
26. Yang, H., Zhang, Y., Zhou, Y., Fu, X., Liu, H., & Vasilakos, A. V. (2014). Provably secure three-party authenticated key agreement protocol using smart cards. *Computer Networks*, *58*, 29–38.
27. Yao, G., Bi, J., & Vasilakos, A. V. (2015). Passive IP traceback: Disclosing the locations of IP spoofers from path backscatter. *Information Forensics and Security, IEEE Transactions on*, *10*(3), 471–484.
28. Zeng, Y., Xiang, K., Li, D., & Vasilakos, A. V. (2013). Directional routing and scheduling for green vehicular delay tolerant networks. *Wireless Networks*, *19*(2), 161–173. doi:10.1007/s11276-012-0457-9.
29. Zhou, J., Dong, X., Cao, Z., & Vasilakos, A. V. (2015). Secure and privacy preserving protocol for cloud-based vehicular DTNs. *IEEE Transactions on Information Forensics and Security*, *10*(6), 1299–1314.
30. Zhu, H., Lin, X., Lu, R., Fan, Y., & Shen, X. (2009). Smart: A secure multilayer credit-based incentive scheme for delay-tolerant networks. *Vehicular Technology, IEEE Transactions on*, *58*(8), 4628–4639.

Sujoy Saha is presently a faculty member in the Department of Computer Applications, National Institute of Technology, Durgapur, India. He has completed his Ph.D. from NIT Durgapur in 2015, M.Tech. in Computer Science and Engineering from Jadavpur University in 2005, and B.Tech. in Computer Science and Engineering from NIT Calicut. His areas of research are Mobile Ad-hoc Networks, Network Modeling, Delay Tolerant Networks and Network Security. He has published 11 papers in different International Conferences/Journals which include leading conferences like ACM Mobicom, ACM DEV, ICDCN, COMSNETS, PERCOM, INFOCOM, etc.

Subrata Nandi is presently working as an Associate Professor in the Department of Computer Science and Engineering, NIT, Durgapur, India. He completed his B.Tech. in Computer Science & Engineering from University of Calcutta and M.Tech. in Computer Science & Engineering from Jadavpur University. He completed his Ph.D., titled 'Information Management in Large Scale Networks' from the Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur in 2011. He worked as a project fellow in the Indo-German (DST-BMBF) project during 2008–2010. He visited Dept. of High Performance Computing, TU Dresden, Germany, as visiting research

scientist and received ACM SIGCOMM travel grant in 2008. He has published 25 papers in different International Conferences/Journals, which include Physical Review E, ACM MobiCom, ACM SIGCOMM, ACM DEV, etc. His broad interests lie in developing technologies for developing regions with specific focus on Peer to Peer Network, Mobile Ad-hoc Network, Delay Tolerant Network, Service-oriented Architecture, etc.

Rohit Verma is a Ph.D. Scholar in the Department of Computer Science and Engineering at Indian Institute of Technology Kharagpur, India. He received his B.Tech. in the Department of Computer Science and Engineering from National Institute of Technology, Durgapur, India in 2013. His research interests include computer systems, mobile computing and delay tolerant networks. He has several international conference publications, such as in IEEE INFOCOM and MobiSys Workshop.

Satadal Sengupta is a Master of Science (MS) by Research student and a Junior Research Fellow in the Department of Computer Science and Engineering at Indian Institute of Technology Kharagpur, India. He is currently a part of the Complex Networks Research Group (CNeRG) and works with Dr. Sandip Chakraborty and Prof. (Dr.) Niloy Ganguly. His research interests lie in the fields of Mobile Computing and Wireless Networks. Satadal completed his Bachelor of Technology in Computer Science and Engineering from National Institute of Technology Durgapur, India in 2013. He worked as an Associate Applications Developer in Oracle Financial Services Software Ltd., Bangalore, India from September, 2013 to April, 2015. Satadal has also worked with Microsoft IT, Hyderabad, India as a Software Development Engineer (SDE) during his summer internship from May to July, 2012.

Kartikeya Singh is presently working as a Senior Software Developer at HiveMinds Innovative Market Solutions. He has completed his B.Tech. from NIT Durgapur in Computer Science and Engineering by 2014. His areas of research are Real Time Big Data Analysis, Delay Tolerant Networks and Network Security, Rule Engine Design, etc.

Vivek Sinha is presently working as a Software Development Engineer II at Via.com. He has completed his B.Tech. from NIT Durgapur in Computer Science and Engineering, in 2014. His areas of research and work are Data mining - which includes Big Data Analysis and Machine Learning, Delay Tolerant Networks, and Network Security.

Sajal K. Das is the chair of the Computer Science department and the Daniel St. Clair Endowed Chair Professor at the Missouri University of Science and Technology (MST). He has completed B.S. Degree in Computer Science from Calcutta University in 1983 and M.S. degree in Computer Science from Indian Institute of Science in Bangalore in 1984. He has completed Ph.D. degree in Computer Science from University of Central Florida in 1988. During 2008–2011, he served the US National Science Foundation as a Program Director in the division of Computer Networks and Systems. His research interests include wireless and sensor networks, mobile and pervasive computing, smart environments and smart health care, pervasive security, biological networking, applied graph theory and game theory. Das has published over 650 papers, gathering 15,500+ citations according to Google Scholar, and 50 invited book chapters. He holds 5 US patents, coauthored 51 book chapters and four books titled Smart Environments: Technology, Protocols, and Applications (2005), Handbook on Securing Cyber-Physical Critical Infrastructure: Foundations and Challenges (2012), Mobile Agents in Distributed Computing and Networking (2012), and Principles of Cyber-Physical Systems (2016). His h-index is 72 with more than 20,500 citations according to Google Scholar. Dr. Das received 10 Best Paper Awards in such prestigious conferences as ACM MobiCom'99, IEEE PerCom'06 and IEEE SmrtGridComm'12.