Egyptian Informatics Journal 24 (2023) 205–214

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

TBMOR: A lightweight trust-based model for secure routing of opportunistic networks

Bing Su [*], Ben Zhu

School of Information & Engineering, Changzhou University, Changzhou, China

A R T I C L E I N F O

Article history:
Received 27 July 2022
Revised 4 December 2022
Accepted 21 February 2023
Available online 10 March 2023

Keywords:
Opportunistic networks
Trust model
Routing algorithm

A B S T R A C T

Opportunistic networks are self-organizing dynamic networks that use the communication opportunities presented by node movement to enable data delivery. Due to the indirect nature of inter-node connections and dynamic topology, opportunistic networks are vulnerable to malicious attacks, which poses a great challenge to their security. In this paper, we first propose a trust-based model for the secure routing of opportunistic networks (TBMOR), which incorporates forwarding positive degree (FP) and node activity degree (NA) into the calculation of trust value, evaluates nodes in the network by a dynamic combination of direct trust and indirect trust, and then optimizes the next node list using trust values and pruning strategies. Secondly, this paper extends the network life cycle by dynamically setting up energy thresholds to reduce network redundancy by dynamically partitioning the number of replicas using trust values.Simulation results show that compared to other routers, TBMOR can effectively resist malicious attacks, has a higher message delivery rate and lower transmission latency than other routers.

1. Introduction

With the continuous development of wireless communication technology in recent years, many smart devices have appeared in people's work and life, and the wide application of wireless technology in various fields has led to a surge in the demand for bandwidth, coverage, and reliability [1]. Traditional self-organizing networks usually determine a complete end-to-end link in advance before communication; however, in many scenarios, many wireless devices move frequently and are unevenly distributed, which leads to frequent communication interruptions in traditional self-organizing networks and makes it difficult for devices to communicate with each other properly [2]. Since traditional self-organizing networks cannot adapt well to the changes of a dynamic wireless environment, the notion of opportunistic networks is gradually emerging [3].

An opportunistic network makes full use of wireless transmission characteristics, transmits data one by one through the communication opportunities formed by the movement of nodes in the network, and uses the "store-carry-forward" routing mode to realize the communication between nodes [4]. Currently, opportunistic networks have been used in wildlife research [5], in-vehicle networks [6], and post-disaster communication [7].

However, due to the openness and mobility of opportunistic networks, the networks are vulnerable to attacks by malicious nodes, which not only reduces the security of network data transmission but also greatly affects the packet transmission efficiency and increases the network overhead. Therefore, how to ensure that the next candidate node is not a malicious node and how to select a suitable forwarding node from it has become an important research direction in the opportunistic networks field [8].

Existing opportunistic network routing techniques usually aim to improve packet forwarding success and reduce packet forwarding latency in a secure network environment, and do not take measures to identify and defend against malicious attacks, which can cause significant damage to the network if there are malicious nodes in the network. For malicious behavior, traditional MANET networks generally use an encryption mechanism and a reputation-based incentive mechanism to resist malicious attacks; however, this approach is not applicable to opportunistic networks. Firstly, since the nodes in the opportunistic network are always moving, it is difficult to have an authorization center in the network; secondly, the reputation mechanism that monitors the behavior of nodes through the watchdog mechanism in the traditional network is also difficult to apply in the opportunistic network due to the mobility of the nodes in the opportunistic

* Corresponding author.
E-mail address: subing@cczu.edu.cn (B. Su).

https://doi.org/10.1016/j.eij.2023.02.002

network; finally, the computational power, energy, and bandwidth of the nodes in the opportunistic network are very limited, which is not suitable for overly complex routing algorithms [9].

To address the above problems, this paper proposes a lightweight trust-based opportunistic routing (TBMOR) for the secure routing of opportunistic networks. To effectively resist malicious attacks, TBMOR calculates the direct trust value by the number of node historical interactions, historical interaction objects, and historical forwarded messages, and calculates the indirect trust by considering the recommended values of neighboring nodes, and finally dynamically assigns the weights of direct trust and indirect trust to form a comprehensive trust value. Firstly, the alternative forwarding nodes are initially screened by the comprehensive trust value to form the next node list; then, the list is optimized by using a trust value and pruning strategy, and finally, the node with a higher trust value is selected for forwarding. Each node maintains a trust table and a message completion record table, and the nodes update the trust table after the interaction. After receiving the message, the node records the message ID in the message completion record table and passes it to the next node. After receiving the record table, the next node updates its own record table immediately and then checks if there is a packet in the cache that has been forwarded, deleting it if it exists. This strategy can greatly reduce the number of redundant packets in the network, improve the packet forwarding rate, and reduce the network forwarding delay. networks routing algorithm. The contributions of this research are as follows:

1. A lightweight trust-based secure routing model for opportunistic networks is proposed for possible malicious attacks in opportunistic networks;
2. In the forwarding phase, low-energy nodes are filtered out by dynamically setting energy thresholds to extend the network life cycle. A limited replica policy is used to dynamically partition replicas using trust values, which reduces network redundancy and improves network performance;
3. The simulation results show that TBMOR can effectively resist malicious attacks and has a higher delivery rate and lower transmission delay than other routing algorithms.

The rest of this paper is organized as follows. Some background and current research work conducted for OppNets routing and trust model are presented in Section 2. Section 3 presents the proposed TBMOR trust model and gives the algorithm flow. Section 4 introduces the routing forwarding strategy in detail and gives the algorithm flow. Section 5 contains three simulation experiments to explore the impact of packet lifetime, node caching, and malicious attacks on routing algorithms. Section 6 concludes the paper.

2. Related work

2.1. OppNets routing

Opportunistic networks take advantage of the encounter opportunities brought about by node movements and use the "store-ca rry-forward" data forwarding mode for data delivery. Since the nodes in the opportunistic network are mobile, sparse, and resource-limited, how to select the next node for reasonable packet delivery between nodes and improve the success rate of packet forwarding has become one of the key points of opportunistic network routing research. At present, scholars have proposed many opportunistic network routes, which can be divided into copy-based routing, active motion-based routing, and utility-based routing, according to the transmission strategy [10].

Copy-based routing increases the packet delivery rate by increasing the redundancy of packets in the network, and a typical routing algorithm is Epidemic [11]; Epidemic uses a flooding mechanism in the network, and the node-carrying packets will make a copy of its own packets and pass them out when it meets other nodes; this infinite copy method obviously improves the delivery rate and reduces the transmission delay, and at the same time, it also leads to the existence of a large number of redundant copies in the network, which results in the waste of network resources. Therefore, some scholars have proposed the opportunistic network routing Spray and Wait [12], which fixes the number of copies of each packet in the network and divides the packet delivery process into two phases. In the spray phase, the node will make a copy of the packet every time it meets a new node and forward it to the new node; when the number of remaining copies of the packet is 1, the node carrying the packet will enter the wait phase, in which no copy of the packet will be generated and the node will not forward the packet until it encounters the destination node.

Active motion-based routing improves the delivery rate by introducing some nodes that can move actively in the network, and a typical routing algorithm is the Message Ferrying algorithm [13], which deploys a ferry node in the network. Ferry nodes move according to a predetermined schedule to improve packet delivery rates through proactive delivery. Due to the limited caching and energy of a single ferry node, the smaller network life cycle may occur in a more complex network. Ref. [14] proposed the form of multiple ferry nodes to improve network reliability and transmission efficiency.

Utility-based routing introduces a utility function to evaluate the utility value of next-hop nodes, which avoids the blind delivery of packets. Prophet routing [15] improves the packet delivery rate by calculating the forwarding probability to predict the selection of next-hop nodes, so that packets can be delivered from low-probability nodes to high-probability nodes. Bubble Rap [16] divides the community and selects the nodes with the highest current community activity nodes with the highest community activity for forwarding, thus improving the delivery rate.

2.2. Trust model

In order to deal with the impact of malicious and private nodes on the security and stability of opportunistic network, many scholars have proposed different security solutions. Among them, cryptography and trust models are the most effective technologies to implement secure routing. However, the encryption mechanism is expensive and difficult to deploy in opportunistic networks. The trust model has become the focus of the research on the secure routing of opportunistic networks.

The earliest application of trust models in networks can be traced back to Beth et al. [17], who proposed calculating the trust of a node by computing the arithmetic mean of the recommended trust values provided by other nodes. In the model, which has been heavily cited by researchers today, they divided trust into direct trust and indirect trust.

Eric et al. [18] proposed a trust framework for opportunistic mobile social networks. The framework derives a new trust quantification method based on three factors: satisfaction, fitness, and connectivity, and proposes a "two-hop feedback method" to enable nodes to receive timely acknowledgements. Simulation results show that the trust framework can effectively resist malicious attacks. Li et al. proposed a secure routing decision algorithm based on a trust mechanism [19] that calculates the trust value of intermediate nodes and stores it in a vector table by collecting the packet forwarding evidence chain as well as message delay from the target node; the forwarding evidence of intermediate nodes is bound in the packets during the delivery process through a digital signature mechanism, and finally, is passed out by means of a trust gradient.

In the literature, Kandhoul et al. [20] proposed a trust-based security framework for OppIoT called T_CAFE, which calculates trust values based on parameters such as CPR, FR, and ER and sets thresholds for these parameters to identify malicious attacks—if the FR value between two nodes is greater than the threshold, then it is identified as a black hole attack and isolated; if the ER value between two nodes is greater than the given threshold, then it is identified as a witch attack. This protocol can better resist malicious attacks and maintain a high delivery rate. In [21], in which an energy efficient trust management and routing mechanism is constructed for selective forwarding attacks and new-flow attacks in the network, a lightweight behavior monitoring and evaluation scheme is implemented by extending the flow table of nodes, and subsequently, a centralized trust management mechanism is proposed to detect and isolate malicious nodes using the collected information. The protocol integrates the trust and energy levels of nodes to improve the delivery rate and extend the network lifetime.

Waleed et al. [22] proposed a prediction-based trust model that uses historical behavior to predict the future behavior of nodes, and experiments show that the model performs well even in highly congested networks. A trust model based on node behavior was proposed by Su et al. [23]. The trust model uses pruning and filtering mechanisms to remove malicious suggestions, and a dynamic weight calculation method is used to combine direct and indirect trust in calculating the integrated trust value, which can filter low-trust nodes in the network. Then, a trust model-based opportunistic routing algorithm (BTOR) is proposed by combining ETX (expected transmission count) values and node trust values.

Samaneh et al. [24] proposed a secure architecture for opportunistic network routing where nodes can decide to carry messages without knowledge of the sender, destination, and intermediate nodes. In the case when all nodes in the network are not real, the model proposes a trust structure, a collaborative approach, and a method for sharing public keys. In addition, a routing algorithm is introduced to select the best available nodes for forwarding messages. This routing algorithm provides identity and location privacy for all nodes and guarantees message confidentiality and data integrity. This trust model can effectively resist selfish behavior and black hole attack attacks and can prevent Sybil attacks. However, the introduction of encryption methods in the trust model also intensifies the resource consumption of nodes.

Wu et al. [25] proposed an edge collaboration cache trust community routing algorithm (ECTC), which is documented in the context of opportunistic community networks and mainly uses node similarity to calculate trust values. The local trust community modules are then partitioned to achieve efficient community retrieval. Simulation experimental results show that the edge collaborative caching trust community routing algorithm (ECTC) can significantly improve the delivery efficiency and reduce the network overhead. Bangotra et al. [26] proposed a trust-based secure intelligent opportunistic routing protocol (TBSIOP) that uses sincerity in forwarding data packets (Fs), sincerity in acknowledgment (ACKs), and energy depletion (Ed) as metrics for calculating trust values, and then uses the trust values to prevent malicious nodes from being selected as relays through a relay selection algorithm. Compared with other algorithms, this algorithm improves the network delivery rate and network lifetime.

3. Trust Model

In the opportunistic network scenario, nodes move frequently and node density is relatively sparse. Centralized algorithms are very difficult to implement in this network environment, so distributed algorithms will be used in this paper.

In this paper, a combination of direct trust and indirect trust is used to comprehensively evaluate the trustworthiness of nodes. Direct trust is calculated based on the direct interaction behavior of both nodes and the historical interaction behavior of the evaluated node. Since a malicious node in the network may misrepresent its own interaction information to deceive the evaluation node to obtain a high trust value, it is necessary to consider the evaluation of the node by other nodes. Indirect trust is the trust value calculated by the valuation node, considering the trust degree of the evaluated node's neighboring nodes to the evaluated node. Finally, the combined trust value is calculated by dynamically assigning the weights of both.

3.1. Direct trust

Direct trust is the direct perception of the evaluation node to the evaluated node. In this paper, the following five factors are considered as trust factors for the direct trust calculation, where i denotes trust in the evaluated node and j denotes the node to be evaluated:

1. Bayesian Trust Degree (BT)
The Bayesian trust model treats the trust degree as a random variable obeying a probability distribution (α distribution$f(x;\alpha,\beta)$), and infers the future nodes' behavior (posterior) from the nodes' historical interaction information (prior). The number of successful interactions is taken as the parameter α of the β distribution, and the number of failed interactions is taken as the parameter β of the β distribution; then, the expectation of this β score can be used as the Bayesian trust degree of the node [27]. The Bayesian trust degree can visually reflect the success rate and trend of the node in forwarding packets, and the higher the value, the greater the ability of the node to forward packets successfully. Its calculation formula is as follows:
$$BT_{ij} = \frac{\alpha}{\alpha+\beta} = \frac{n_s+1}{n_s+n_u+2} \tag{1}$$
where n_s represents the record of successful j interactions and n_u represents the record of failed j interactions;
2. Forwarding Positive Degree (FP)
Forwarding aggressiveness indicates how aggressive a node is about packet forwarding, which is determined by the packet forwarding delay. Its higher value means that the node is more willing to participate in packet forwarding. Its calculation formula is as follows:
$$FP_{ij} = e^{-\lambda\times\sum_{x=1}^{m}\frac{MsgDelay_x}{MsgTTL_x}} \tag{2}$$
where m denotes the number of packets that have been forwarded by node j; $MsgDelay_x$ denotes the time interval between packet being accepted by node j and forwarding; $MsgTTL_x$ represents the survival period of packet x; and λ is the adjustment factor, whose smaller value means that the trust value is more sensitive to the forwarding delay;
3. Data Similarity Degree (DS)
Data similarity indicates the degree of similarity between two nodes forwarding data; nodes in the same space have similarity in forwarding tasks, and taking similarity into account in the trust value calculation can resist malicious attacks to some extent. It is calculated as follows:
$$DS_{ij} = \frac{MsgSame}{(List_i+List_j)/2} \tag{3}$$

where *MsgSame* denotes the number of similar packets between node *i* and node *j* and $(List_i + List_j)/2$ denotes the average number of packets in the cache of both nodes;

4. Node Activity Degree (NA)
Node activity indicates how active a node is in the whole network, and it is determined by the number of nodes encountered per unit time. The more nodes encountered per unit time, the more active this node is in the network, which means this node has a higher probability of encountering the destination node. In order to resist malicious attacks, node activity is zeroed every unit time, historical node activity is considered at the beginning of the unit time, and the percentage of node activity in the current time interval becomes larger as time grows. Its calculation formula is as follows:

$$NA_{ij} = \frac{t}{\theta} \times \frac{InterNum}{Nodes} + (1 - \frac{t}{\theta}) \times NA_{ij}\_old \tag{4}$$

where θ is the unit time; Δt is the time interval since the last reset; *InterNum* represents the number of nodes encountered in the network from the last reset time to the current time node (no duplication); *Nodes* is the total number of nodes in the current network; and NA_{ij}_old represents the node activity at the last reset;

5. Node connection degree (NC)
The node connectivity indicates the frequency of interaction between node *i* and node *j*, which is determined by the interaction history between nodes. The calculation formula is as follows:

$$NC_{ij} = \frac{Inter2i}{SumInter} \tag{5}$$

where *Inter2i* denotes the number of historical interactions between node *i* and node *j* and *SumInter* denotes the total number of interactions of node *j*.

Therefore, the direct trust value *DT* can be expressed as

$$DT_{ij} = w_1 \times BT_{ij} + w_2 \times FP_{ij} + w_3 \times DS_{ij} + w_4 \times NA_{ij} + w_5 \times NC_{ij} \tag{6}$$

The parameters w_1, w_2, w_3, w_4, and w_5 are the weight coefficients; different values can be assigned to the weights in different network environments, e.g., a higher value can be assigned to w_4 if more attention is paid to node activity.

3.2. Indirect trust

Similar to social life, if node *i* wants to evaluate node *j* more comprehensively, it should consider the “view” of neighboring nodes on *j*. Suppose there are *n* common neighbor nodes between nodes *i* and *j*. In order to exclude the influence of malicious nodes on the indirect trust degree calculation, the common neighbor nodes need to be filtered again before the calculation.

1. Define the trust threshold and delete the nodes below the trust threshold.
First, filter out the nodes that never participate in forwarding, because these nodes are likely to be malicious nodes. Secondly, filter out nodes with low trust values, where the trust threshold is defined as the average trust value of the trust table of node *i*.

$$T_{threshold} = \frac{\sum_{x=1, x\neq i}^{n} T_{ix}}{n-1} \tag{7}$$

where *n* denotes the current number of common nodes and T_{ix} denotes the direct trust value of node *i* to its common neighbor nodes;

2. Node *i* prefers to trust the recommended values of neighboring nodes that interact frequently with node *j*.
After (1), there are *m* co-neighboring nodes left. The “opinion” from the node that interacts with node *j* relatively frequently should be more informative, so we use the parameter σ to represent the deviation of the number of interactions between neighboring nodes and node *j* to the number of interactions between node *i* and node *j*, which is calculated as follows:

$$\sigma = \sqrt{\frac{\sum_{k=1}^{m} (CN_{k,j} - CN_{i,j})^2}{m}} \tag{8}$$

where $CN_{k,j}$ represents the number of interactions between neighbor node *k*. If $CN_{kj} < CN_{ij} - \sigma$, then the “view” of node *k* will be considered invalid.

Eventually, *p* common neighbor nodes remain; then, the final indirect trust calculation formula is

$$IT_{ij} = \frac{\sum_{x=1}^{p} DT_{i,x} \times DT_{x,j}}{p}. \tag{9}$$

3.3. Comprehensive trust

The final formula for calculating the comprehensive trust value by combining direct trust and indirect trust is as follows:

$$T_{ij} = e^{-\mu \times N} \times DT_{ij} + (1 - e^{-\mu \times N}) IT_{ij} \tag{10}$$

Among them, $e^{-\mu \times N}$, $(1 - e^{-\mu \times N})$ is the weighting factor; *N* is the number of common neighbors; μ is the moderating factor, and the larger its value the smaller the influence of *N*, and the greater will be the weight of the direct trust value. In addition, this paper defines the decay formula for the trust value as follows:

$$T_{ij} = T_{ij}\_old \times (1 - \frac{t_n - t_0}{t_n})^{\gamma} \tag{11}$$

where γ is the attenuation factor. The entire trust model is described in Algorithm 1.

Table 1
Simulation environment setting.

Simulation environment parameters	Numerical value
Scenario.updateInterval	0.1s
Scenario.endTime	86400s
MovementModel.worldSize	4500 m × 3400 m
MovementModel.warmup	1000s

Table 2
Node setting.

Node Parameters	Numerical value
Group.nrofHost	200
Group.movementModel	ShortestPathMapBasedMovement
Group.bufferSize	5 M ~40 M
Group.interface1	btInterface
btInterface.transmitRange	10 m
btInterface.transmitSpeed	250 k

Algorithm 1: Trust Model

```
Input: $N_i$:Trustor;
$\lambda$:the adjustment factor of Forwarding Positive Degree ;
$\theta$:the unit time of Node Activity Degree;
$\omega_1, \omega_2, \omega_3, \omega_4, \omega_5$: the weight coefficients and $\omega_1 + \omega_2 + \omega_3 + \omega_4 + \omega_5 = 1$;
$\mu$:the moderating factor of Comprehensive trust
Share $deliveredMsg\_List$ among nodes upon encounter;
foreach $Msg$ of $DeliveredMsg\_List$ do
    if $MsgForward\_list$ of $N_i$ has $Msg$ then
        Delete $Msg$ from $MsgForward\_list$ of $N_i$;
    end
end
for all $N_j \in NeighborNode$ of $N_i$ do
    Initial $BT_ij = FP_ij = DS_ij = NA_ij = NC_ij = 0$;
    if $N_j.InterNum + N_j.InterSucNum! = 0$ then
        Compute Bayesian Trust Degree as $BT_{ij} = \frac{n_s+1}{n_s+n_u+2}$
    end
    if $N_j.MsgDelayTable.size! = 0$ then
        Compute Forwarding Positive Degree as
        $FP_{ij} = e^{-\lambda \times \sum_{x=1}^{m} \frac{MsgDelay_x}{MsgTTL_x}}$
    end
    if $N_j.MsgList.size + N_j.MsgList.size! = 0$ then
        Compute Data Similarity Degree as $DS_{ij} = \frac{MsgSame}{(List_i + List_j)/2}$
    end
    if $CurrentTime - LastUpdateTime > 0$ then
        Compute Node Activity Degree as
        $NA_{ij} = \frac{t}{\theta} \times \frac{InterNum}{Nodes} + (1 - \frac{t}{\theta}) \times NA_{ij}\_old$
    else
        $NA_ij = lastNA_ij$
    end
    if $N_j.SumInterNum! = 0$ then
        Compute Node Connection Degree as $NC_{ij} = \frac{Inter2i}{SumInter}$
    end
    Compute $DirectTrust$ as
    $DT_{ij} = w_1 \times BT_{ij} + w_2 \times FP_{ij} + w_3 \times DS_{ij} + w_4 \times NA_{ij} + w_5 \times NC_{ij}$;
    $N_i$ invites $CommonNeiborNodes$ between $N_i$ and $N_j$;
    Compute $TrustThreshold$ as $T_{thre} = \frac{\sum_{x=1}^{n} T_{in}}{TN}$;
    forall $N_x \in CommonNeiborNodes$ do
        if $T_ix < T_thre$ then
            Delete $N_x$ from $CommonNeiborNodes$
        end
    end
end
Compute $IndirectTrust$ as $IT_{ij} = \frac{\sum_{x=1}^{p} DT_{i,p} \times DT_{x,p}}{p}$;
Compute $TrustValue$ as $T_{ij} = e^{-\mu \times N} \times DT_{ij} + (1 - e^{-\mu \times N}) IT_{ij}$;
Output: $TrustValue$
```

4. Forwarding strategy

The forwarding strategy considers two main aspects, the trust value of the alternative nodes and the energy level of the nodes. The node moves through the network with packets and, at a fixed interval, uses the trust model to evaluate the trust of the nodes within communication range and updates the trust table in its own cache. The node then adds the neighboring nodes to the list of alternative forwarding nodes, and the algorithm first considers the trust value of the nodes in the alternative forwarding list, defining T_{th} as the trust threshold of the list nodes, with the following equation:

$$T_{th} = \sqrt{\frac{\sum_{x=1}^{n} T_{ix}^2}{n}} \tag{12}$$

The root mean square value of the trust value in the list is used here as the trust threshold, where T_{th} denotes the trust threshold in the alternative forwarding nodes, n denotes the number of nodes in the list, and T_{ix} denotes the trust value of node i to the nodes in the alternative forwarding list. Nodes with a trust value less than T_{th} will be removed from the alternative forwarding list.

Then, the node adds neighboring nodes to the list of alternative forwarding nodes and the algorithm further refines the list considering the energy level of the nodes.

$$E_{\text{Threshold}} = \frac{n}{n+1} \times \bar{E} \tag{13}$$

where $E_{\text{Threshold}}$ denotes the energy threshold, n is the number of nodes in the alternative forwarding list, and $\bar{E}$ is the average value of the remaining energy of the nodes in the alternative forwarding list; from the formula, we can see that when there are more nodes, the algorithm will set the threshold value relatively high, and when there are fewer nodes, the threshold value will be set very low. Nodes above the energy threshold value will be added to the final alternative forwarding list and sorted according to their trust values, waiting for nodes to be forwarded.

In order to reduce the degree of redundancy of messages in the network, this paper adopts a limited copy forwarding strategy. The limited copy forwarding strategy not only increases the probability of successful packet forwarding to a certain extent, but also reduces the number of copies in the network, reduces the network load, and improves the network throughput.

Let node i, carrying a message with ID x, encounter node j in the network to be forwarded; the number of copies of message x that is carried by node i is $ND_{i,x}$ (ND, number duplicate), and the destination node of message x is k. The copy allocation strategy of this paper is:

1. If $T_{ik} > T_{jk}$, then $ND_{j,x} = 1$ and $ND_{i,x} = ND_{i,x} - 1$;
2. If $T_{ik} \leqslant T_{jk}$, then $ND_{j,x} = \lceil ND_{i,x}/2 \rceil$, and $ND_{i,x} = \lfloor ND_{i,x}/2 \rfloor$.

The node will exchange the message completion table with the sender and update the table while receiving the message, and then immediately check whether its cache contains the message ID recorded in the table, and if so, delete it in time; after that, it will determine whether it is the destination node of the message, and if so, it will add the message ID to the message completion table and pass it out in the next interaction, so that the network nodes will be aware of the messages that have reached the destination node and delete the redundant messages in the cache. The complete forwarding strategy is shown in Algorithm2.

Algorithm2: Forwarding Strategy

```
   Input: $CandidateForwardNode\_List$:List of candidate forwarding
          nodes $DeliveredMsg\_List$: List of messages to be forwarded
 1 Initialize $Forward\_List$;
 2 foreach $message\ of\ DeliveredMsg\_List$ do
 3     if the destination node of the message is in the
        $CandidateForwardNode\_List$ then
 4         start transfer;
 5     end
 6 end
 7 compute Energy_Threshold as $E_{\text{Threshold}} = \frac{n}{n+1} \times \bar{E}$;
 8 foreach $Node\ of\ CandidateForwardNode\_List$ do
 9     if $Node\_Energy < Energy\_Threshold$ then
10         delete this Node from $CandidateForwardNode\_List$;
11     end
12 end
13 for all $Msg \in DeliveredMsg\_List$ do
14     foreach $N_j \in CandidateForwardNode\_List$ do
15         if $N_j$ has not $Msg$ and $N_j$ is not transferring then
16             if $T_i x > T_j x$ then
17                 $N_i$ divide the number of copies of $Msg$ by half to $N_j$
18             else
19                 $N_i$ will give $N_j$ 1 copy of $Msg$
20             end
21             $Forward\_List$.add($N_j$,$Msg$);
22         end
23     end
24 end
   Output: $Forward\ \ List$
```

5. Simulation

This paper uses the ONE (opportunistic network environment) simulator to simulate TBMOR. The ONE simulator is a simulation tool based on Java environment specially developed for opportunistic networks. The simulator provides a variety of movement models and report types, as well as many classical algorithms. In this paper, the Epidemic, Prophet and Spray and Wait algorithms are alternatively selected as control algorithms and compared with the TBMOR algorithm in the ONE simulator environment to verify the superiority of the TBMOR algorithm.

5.1. Simulation environment

In this paper, the simulation map is a part of the real map of the city of Helsinki brought by the ONE simulator; the simulation area is 4500 m $\times$ 3400 m, and the simulation time is 24 h. The normal node randomly generates packets of size 512–1024 k in the time interval of 25–35 s, and its survival period is 30–300 min with cache size 5–40 M. To reduce the error, we adjusted the random factor of the motion model for five simulations. The adjustment factor γ of forwarding positivity in the trust model is set to 0.025, and the node activity The value of unit reset time θ is 3600 s; the adjustment factor μ for calculating the integrated trust value is set to 0.15, and the trust decay factor γ is 1.0. The specific parameters of the simulation are shown in Tables 1–3.

Table 3
Message setting.

Simulation environment parameters	Numerical value
Events1.class	MessageEventGenerator
Events1.interval	25 s–35 s
Events1.size	512 k–1024 k
Group.msgTtl	30 min–300 min

5.2. Simulation result

In this section, we will observe the performance of each routing in three scenarios. In the first scenario, we will set different message lifetime sizes to observe their impact on the routing protocol; the cache size largely determines the ability of a node to carry the number of packets. An algorithm that performs better with the same cache makes better use of the cache and has a greater ability to deliver packets. Therefore, in the second section, we will adjust the node cache size to analyze the impact of cache size on the performance of the routing protocol; in the last scenario, we will inject different numbers of point malicious nodes into the network to examine the impact of malicious attacks on the routing protocol and verify the effectiveness of TBMOR against malicious attacks.

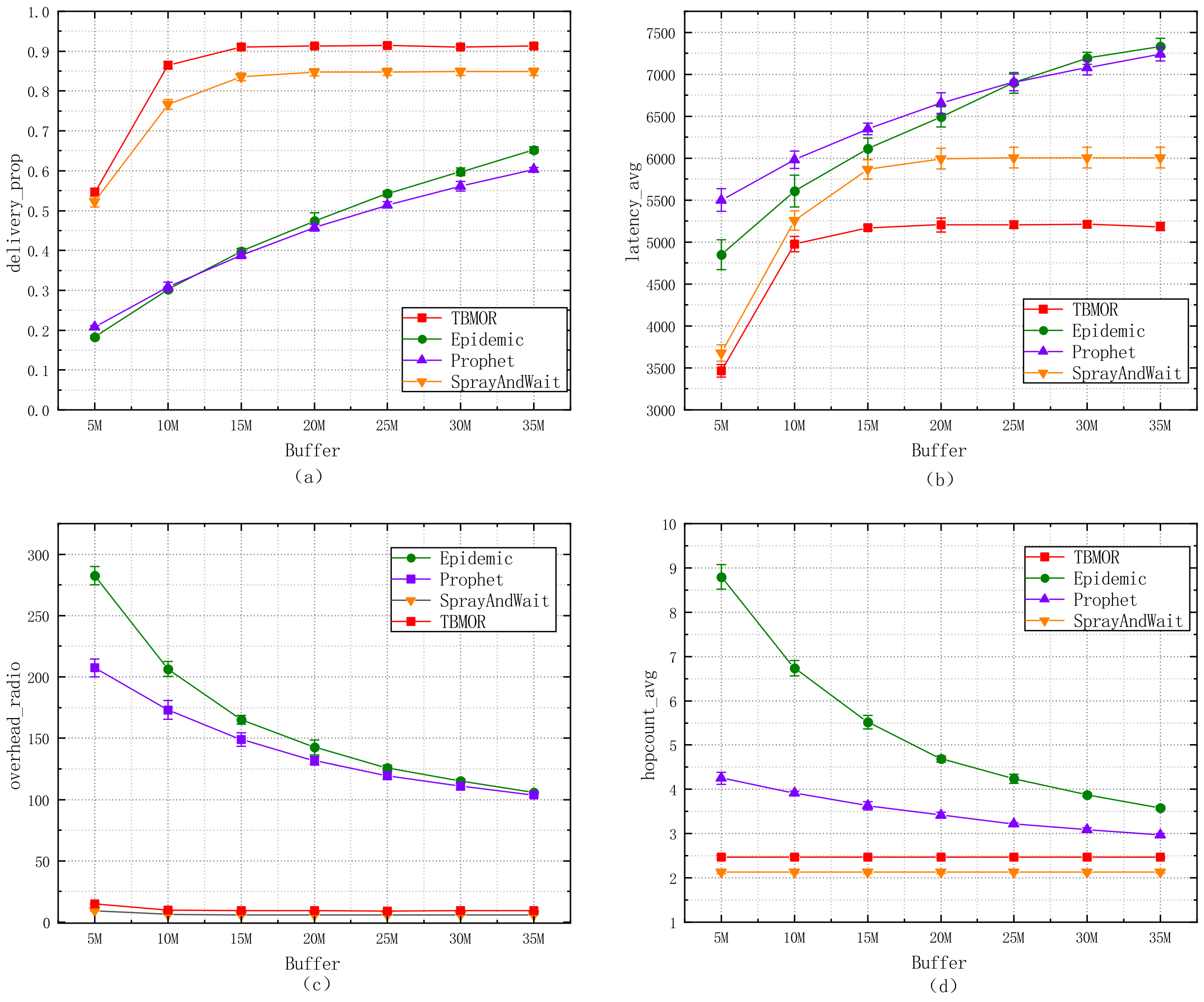

Fig. 1. The impact of node cache size on the performance of routing algorithms. (**a**) Impact of cache on *delivery_prop*; (**b**) impact of cache on *latency_avg*; (**c**) impact of cache on *overhead_radio*; (**d**) impact of cache on *hopcoumt_avg*.

To evaluate the performance of routing protocols, this paper will be based on the following four metrics.

- **The average transmission delay** (*latency_avg*): *latency_avg* indicates the average time required for a packet to finally reach the destination node from the source node, and its smaller value indicates that the transmission capability of the algorithm is stronger;
- **Successful transmission rate**(*delivery_prob*): *delivery_prob* is the ratio of the number of successfully transmitted packets to the total number of packets generated, which reflects the ability of the routing protocol to correctly deliver packets and is the most important indicator;
- **The average hop count** (*hopcount_avg*): *hopcount_avg* indicates the number of passes that the packet undergoes from the source node to the destination node, and the smaller the value means that the next-hop node selected by the routing algorithm is more accurate;
- **Routing overhead** (*overhead_ratio*): *overhead_ratio* is the ratio of the number of packets in the network that do not reach the destination node to the number of packets that successfully reach the destination. The smaller the value means there are less redundant packets in the network; it also means that the delivery cost is smaller, so this value is used as an important indicator to evaluate the energy consumption of nodes.

5.2.1. The impact of node cache size on the performance of routing algorithms

The survival period of the messages is set to 300 min, the node cache size is 5 M–35 M, and all other parameters are kept constant. This experiment investigates how the performance of routing is affected as the node cache increases, and the results are shown in Fig. 1.

From Fig. 1a, it can be seen that the delivery_prob of all four routing algorithms increase when the node cache is expanded, due to the fact that as the node cache increases, the nodes are able to carry more packets through the network, thus allowing more packets to be transmitted, which in turn increases the delivery_prob of the Epidemic, Prophet, and Spray and Wait algorithms. TBMOR, however, achieves a delivery_prob of 91.3% when the node cache is 15 M, and the delivery_prob stabilizes as the node cache increases; this indicates that TBMOR has a lower requirement for node cache than the other three routing algorithms.

From Fig. 1b–d, it can be seen that when the node cache increases, latency_avg increases, overhead_ratio gradually decreases, and the hopcount_avg gradually becomes smaller; this is due to the fact that nodes are able to carry more packets to move around the network, giving more chances of survival to those packets that are delivered too late and which should have been discarded, thus extending the packet delivery time and reducing the overhead_ratio and hopcount_avg. The much smaller latency_avg of Spray and Wait and TBMOR, compared to Epidemic and Prophet,

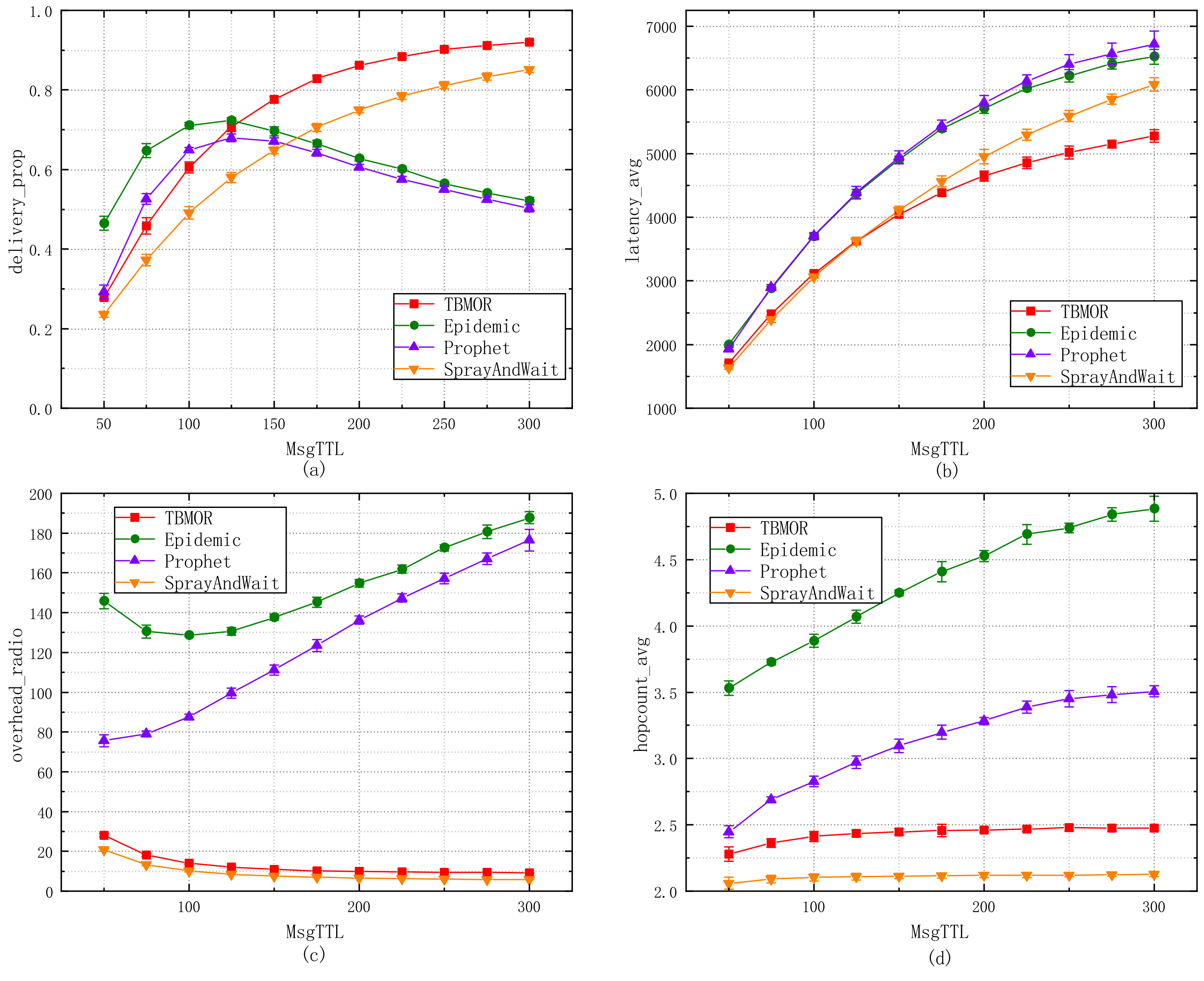

Fig. 2. Impact of packet survival period (TTL) on the performance of routing algorithms.

is due to the fact that these two algorithms use a limited copy strategy, which results in much less packet redundancy in the network, thus allowing valid packets to be forwarded out in a timely manner, which also results in these two algorithms having a smaller overhead_ratio and hopcount_avg. The latency_avg of TBMOR is consistently the smallest of all algorithms and starts to level off at a node cache of 15 M, due to the fact that TBMOR more accurately selects the next node most likely to reach its destination. Although higher than Spray and Wait in terms of overhead_ratio and hopcount_avg, TBMOR significantly outperforms the Spray and Wait algorithm in terms of delivery_prob and latency_avg.

5.2.2. Impact of packet survival period (TTL) on the performance of routing algorithms

The packet survival period is taken as a variable with values from 50 min to 300 min, and the cache is set to 20 M to investigate the effect of the packet survival period (TTL) on the performance of the routing algorithm. The experimental results are shown in Fig. 2.

From Fig. 2, we can see that as the packet survival period grows, the delivery_prob of all four routes increases and latency_avg increases; at a TTL of 300, the TBMOR delivery_prob reaches a maximum of 91.14% and the latency_avg is the lowest; the delivery_prob of the Epidemic and Prophet routes peak at a TTL of 125 min. This is due to the fact that these two algorithms are flooding algorithms, which replicate packets uncontrollably and lead to a large number of redundant packets in the network as the TTL grows. Thus, it reduces the delivery_prob, so the routing overhead_ratio of both algorithms increases significantly at the TTL of 175 min. Although Spray and Wait is slightly better than TBMOR in terms of hopcount_avg and overhead_ratio, TBMOR is better than Spray and Wait in terms of delivery_prob and latency_avg.

5.2.3. The impact of the number of malicious nodes on the routing algorithm

Malicious nodes exhibit malicious behavior by dropping packets, and the impact of malicious behavior on each routing algorithm is examined with their number as a variable. The number of malicious nodes ranges from 0% to 50%, with a packet survival period of 300 min and a node cache of 20 M. The experimental results are shown in Fig. 3.

From Fig. 3a,b, it can be seen that as the proportion of malicious nodes increases, the delivery_prob and latency_avg of all four routing algorithms increase. The TBMOR algorithm proposed in this paper is significantly better than the other three routing algorithms in terms of delivery_prob, possessing a delivery_prob of 91.54% at 20% of malicious nodes and 80.31% at 50% of malicious nodes, which is much higher than other routing algorithms. This shows that TBMOR can identify and filter out a large number of malicious nodes in a malicious environment, which serves as a defense against malicious attacks. With the increasing percentage of malicious nodes, the normal nodes in the network also become sparse; therefore, the probability of meeting and delivering packets

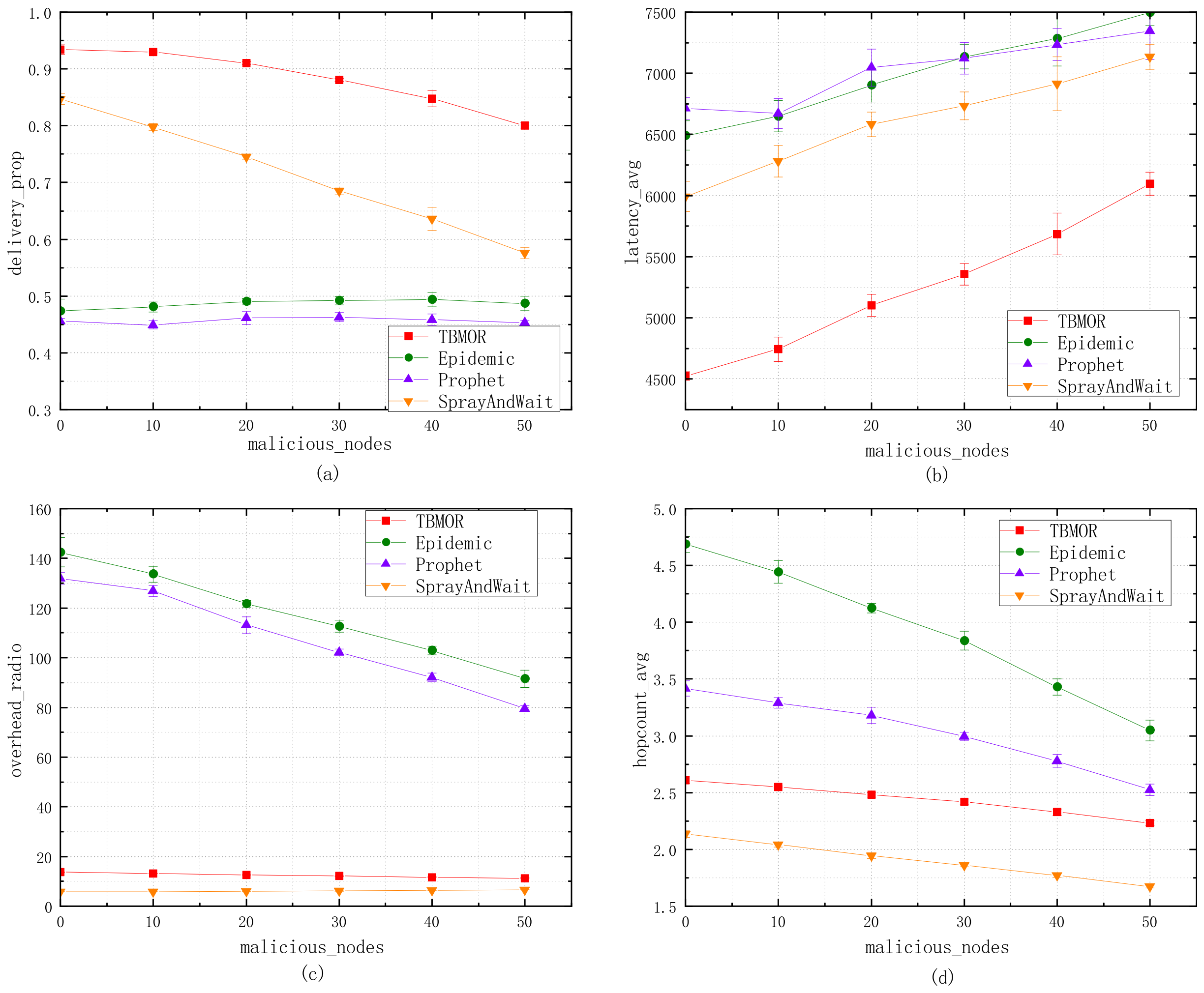

Fig. 3. Impact of the number of malicious nodes on the performance of routing algorithms.

between normal nodes decreases, so this is the main reason for the decrease in the delivery_prob of TBMOR. The other three types of routes have decreased in delivery_prob due to the malicious behavior of malicious nodes, among which Epidemic and Prophet decrease slowly because both routing algorithms are flooding algorithms, and a certain degree of malicious packet loss can instead reduce the redundancy of packets in the network; the packets that may reach the destination by colleagues are maliciously dropped, so the delivery_prob does not change much.

From Fig. 3c,d, it can be seen that in terms of overhead_ratio and hopcount_avg, for Epidemic and Prophet, due to their flooding characteristics, when the proportion of malicious nodes increases, the redundant packets in the network decreases, and their overhead_ratio and hopcount_avg gradually decrease; the Spray and Wait algorithm adopts the limited copy forwarding strategy, so it can maintain a lower overhead_ratio and hopcount_avg. In comparison, the BTMOR algorithm uses a limited-replica policy and a trust model to evaluate nodes, so it can identify and filter malicious nodes. The BTMOR algorithm outperforms other routing algorithms in terms of delivery_prob and latency_avg with a overhead_ratio and hopcount_avg close to that of the Spray and Wait.

6. Conclusions

In this paper, we propose a lightweight trust-based secure routing model for opportunistic networks, where TBMOR calculates the direct trust value of nodes based on the context, indirect trust based on the recommended values of neighboring nodes, and finally, the comprehensive trust value is obtained by the dynamic assignment of weights. In the routing forwarding stage, this paper screens out low-energy nodes by dynamically setting energy thresholds, and dynamically partitions replicas using trust values, which reduces network redundancy and improves network performance. Simulation results show that TBMOR requires a smaller cache than other algorithms, maintains a small network overhead and average hop count even when the packet survival period becomes larger, and has a high delivery rate and low network latency when the network is subject to a widespread malicious attack. In future research, we will consider introducing more complex trust models to ensure the secure communication of nodes, and will pay more attention to the network security in opportunistic social networks and 5G communication environments.

References

[1] Avoussoukpo CB, Ogunseyi TB, Tchenagnon M. Securing and facilitating communication within opportunistic networks: A holistic survey. IEEE Access 2021;9:55009–35. doi: https://doi.org/10.1109/ACCESS.2021.3071309.

[2] S. Singha, B. Jana, S.H. Jana, N.K. Mandal, A survey to analyse routing algorithms for opportunistic network, Procedia Computer Science 171 (2020) 2501–2511, third International Conference on Computing and Network Communications (CoCoNet'19).doi:10.1016/j.procs.2020.04.27.

[3] L. Yu, G. Xu, N. Zhang, F.Q. Wei, Opportunistic network routing strategy based on node individual community, in: 2021 IEEE International Conference on Communications Workshops (ICC Workshops), 2021.

[4] Nayyar A, Batth RS, Ha D-B, Sussendran G. Opportunistic networks: Present scenario- a mirror review. Int J Commun Networks Inf Secur 2018;10.

[5] E.D. Ayele, N. Meratnia, P.J.M. Havinga, An asynchronous dual radio opportunistic beacon network protocol for wildlife monitoring system, in: 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2019, pp. 1–7. doi:10.1109/NTMS.2019.8763854.

[6] Y. Teranishi, T. Kimata, E. Kawai, H. Harai, Hybrid cellular-dtn for vehicle volume data collection in rural areas, in: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), Vol. 1, 2019, pp. 276–284. doi:10.1109/COMPSAC.2019.00048.

[7] Fu X, Yao H, Postolache O, Yang Y. Message forwarding for wsn-assisted opportunistic network in disaster scenarios. J Network Comput Appl 2019;137:11–24. doi: https://doi.org/10.1016/j.jnca.2019.04.005.

[8] Rashidibajgan S, Hupperich T, Doss R, Frster A. Secure and privacy-preserving structure in opportunistic networks. Comput Secur 2021;104(17):102208.

[9] N. Li, S.K. Das, A trust-based framework for data forwarding in opportunistic networks, Ad Hoc Networks 11 (4) (2013) 1497–1509,1. System and Theoretical Issues in Designing and Implementing Scalable and Sustainable Wireless Sensor Networks 2. Wireless Communications and Networking in Challenged Environments. doi: 10.1016/j.adhoc.2011.01.018.

[10] Sachdeva R, Dev A. Review of opportunistic network: Assessing past, present, and future. Int J Commun Syst 2021;34(2).

[11] A. Vahdat, D. Becker, Epidemic routing for partially-connected ad hoc networks, Handbook of Systemic Autoimmune Diseases (2000).

[12] T. Spyropoulos, K. Psounis, C.S. Raghavendra, Spray and wait: An efficient routing scheme for intermittently connected mobile networks, in: Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, WDTN '05, Association for Computing Machinery, New York, NY, USA, 2005, p. 252–259. doi:10.1145/1080139.1080143.

[13] W. Zhao, M. Ammar, E. Zegura, A message ferrying approach for data delivery in sparse mobile ad hoc networks, in: Proceedings of ACM Mobihoc 2004, 2004.

[14] W. Zhao, M. Ammar, E. Zegura, Controlling the mobility of multiple data transport ferries in a delay-tolerant network, in: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 2, 2005, pp. 1407–1418 vol. 2. doi:10.1109/INFCOM.2005.1498365.

[15] Lindgren A, Doria A, Schelen O. Probabilistic routing in intermittently connected networks. In: Dini P, Lorenz P, de Souza JN, editors. Service Assurance with Partial and Intermittent Resources. Berlin Heidelberg, Berlin, Heidelberg: Springer; 2004. p. 239–54.

[16] Hui P, Crowcroft J, Yoneki E. Bubble rap: Social-based forwarding in delay-tolerant networks. IEEE Trans Mob Comput 2011;10(11):1576–89. doi: https://doi.org/10.1109/TMC.2010.246.

[17] T. Beth, M. Borcherding, B. Klein, Valuation of trust in open networks, in:D. Gollmann (Ed.), Computer Security — ESORICS 94, Springer Berlin Heidelberg, Berlin, Heidelberg, 1994, pp. 1–18.

[18] Wang EK, Li Y, Ye Y, Yiu SM, Hui LCK. A dynamic trust framework for opportunistic mobile social networks. IEEE Trans Netw Serv Manage 2018;15 (1):319–29. doi: https://doi.org/10.1109/TNSM.2017.2776350.

[19] Feng LI, Ya-Li SI, Chen Z, Ning LU, Shen LM. Trust-based security routing decision method for opportunistic networks. J Software 2018.

[20] Nisha Kandhoul IW, Dhurandher Sanjay K. T cafe: A trust based security approach for opportunistic iot. IET Commun 2019;13(8):3463–71.

[21] R. Wang, Z. Zhang, Z. Zhang, Z. Jia, Etmrm: An energy-efficient trust management and routing mechanism for sdwsns, Comput Networks 139 (2018), 119–135. doi10.1016/j.comnet.2018.04.009.

[22] Alnumay W, Ghosh U, Chatterjee P. A trust-based predictive model for mobile ad hoc network in internet of things. Sensors 2019;19(6). doi: https://doi.org/10.3390/s19061467.

[23] Su B, Du C, Huan J. Trusted opportunistic routing based on node trust model. IEEE Access 2020;8:163077–90. doi: https://doi.org/10.1109/ACCESS.2020.3020129.

[24] Rashidibajgan S, Hupperich T, Doss R, Forster A. Secure and privacy preserving structure in opportunistic networks. Comput Secur 2021;104:. doi: https://doi.org/10.1016/j.cose.2021.102208102208.

[25] X. Wu, L. Chang, J. Luo, J. Wu, Efficient edge cache collaboration transmission strategy of opportunistic social network in trusted community, IEEE Access PP (99) (2021) 1–1.

[26] D. Bangotra, A. Singh, Y. and Selwal, A trust based secure intelligent opportunistic routing protocol for wireless sensor networks, Wireless Pers Commun (2021). doi: 10.1007/s11277-021-08564-3.

[27] B.E. Commerce, A. Jøsang, R. Ismail, The beta reputation system, in: Proceedings of the 15th Bled Electronic Commerce Conference, 2002.