Stability-based On-demand Multi-path Distance Vector Protocol for Edge Internet of Things

Dongzhi Cao[1], Peng Liang[2], Tongjuan Wu[3], Shiqiang Zhang[1*], and Zhenhu Ning[1]
[1] Faculty of Information Technology, Beijing University of Technology
Beijing, 100124, China
[e-mail: dzcaocwz@126.com, sqzhangbjut@163.com, nzh41034@163.com]
[2] China Center for International Economic Exchanges
Beijing, 100050, China
[e-mail: liangliang20230314@163.com]
[3] Beijing HIWING Scientific and Technological Information Institute
Beijing, 100074, China
[e-mail: wxtbeijing2016@163.com]
[*]Corresponding author: Shiqiang Zhang

Received April 27, 2023; revised July 31, 2023; accepted August 18, 2023; published October 31, 2023

Abstract

In edge computing scenarios, IoT end devices play a crucial role in relaying and forwarding data to significantly improve IoT network performance. However, traditional routing mechanisms are not applicable to this scenario due to differences in network size and environment. Therefore, it becomes crucial to establish an effective and reliable data transmission path to ensure secure communication between devices. In this paper, we propose a trusted path selection strategy that comprehensively considers multiple attributes, such as link stability and edge cooperation, and selects a stable and secure data transmission path based on the link life cycle, energy level, trust level, and authentication status. In addition, we propose the Stability-based On-demand Multipath Distance Vector (STAOMDV) protocol based on the Ad hoc AOMDV protocol. The STAOMDV protocol implements the collection and updating of link stability attributes during the route discovery and maintenance process. By integrating the STAOMDV protocol with the proposed path selection strategy, a dependable and efficient routing mechanism is established for IoT networks in edge computing scenarios. Simulation results validate that the proposed STAOMDV model achieves a balance in network energy consumption and extends the overall network lifespan.

Keywords: Internet of Things, edge computing, trusted path, STAOMDV protocol

1. Introduction

The openness of the Internet of Things (IoT) network poses significant challenges in ensuring secure and efficient data transmission. This is particularly crucial in edge computing scenarios [1], where IoT devices provide external services, necessitating a secure data transmission mechanism to deliver high-quality services. Smart devices such as smartphones and tablets can enhance IoT network performance by relaying data [2]. Therefore, the development of effective routing mechanisms becomes vital to optimize IoT network performance in edge computing scenarios. By addressing security concerns and designing efficient data routing, the full potential of IoT and edge computing can be realized to provide seamless integration and reliable services for a variety of applications.

Edge computing is a paradigm that brings computing and data processing closer to the data sources, reducing latency and network congestion [3]. As a result, IoT devices at the edge can handle data locally, leading to improved response times and enhanced user experience. This distributed architecture not only alleviates the burden on the central cloud infrastructure but also enables real-time data analysis, critical for time-sensitive applications like autonomous vehicles and industrial automation [4]. However, the dynamic and heterogeneous nature of edge environments poses new challenges in terms of network size expansion, device mobility, and resource constraints. Frequent device mobility can lead to link disruptions and topology changes, causing routing paths to become unstable and unreliable [5]. Additionally, the limited energy of IoT nodes in edge scenarios necessitates the development of energy-efficient routing mechanisms to prolong the network's lifespan [6]. Moreover, due to the open nature of IoT networks, they are vulnerable to security threats, such as data tampering and unauthorized access, demanding robust security measures at the routing level [7].

Traditional routing protocols for IoT networks, such as Destination Sequence Distance Vector Routing Protocol (DSDR) [8], Dynamic Source Routing Protocol (DSR) [9], and Ad-hoc On-demand Distance Vector Routing Protocol (AODV) [10], use different criteria to select the routing path. However, they fail to consider the impact of malicious nodes on data forwarding. With the increasing incidence of security issues in IoT networks, security has become an essential consideration in routing algorithm design [11][12]. As a result, trusted routing protocols based on trust have emerged as a research hotspot [13][14]. Such protocols can address security issues associated with data forwarding in IoT networks and ensure the reliability and safety of data transmission [15].

However, the more complex network environment of IoT in edge computing scenarios, such as network size expansion, device mobility, and limited energy of nodes, makes existing trusted routing algorithms less suitable [16]. To address these challenges, a trusted routing mechanism based on edge cooperation is proposed. This mechanism defines and models the link stability attribute between devices based on the possibility of link loss caused by device mobility, energy depletion, and trust relationship changes. Trusted path selection is then achieved based on this stability attribute [17]. The strong data forwarding ability of the edge server is used to avoid selecting excessively long data transmission paths when the source node and destination node are not in the same trust domain. Finally, multi-path routing between nodes is established based on the Ad-hoc On-demand Multi-path Distance Vector (AOMDV) protocol [18][19], and the link stability attributes of paths are collected.

The above challenges and limitations highlight the critical need for a new routing mechanism tailored specifically for edge computing scenarios. The new routing mechanism must not only prioritize secure data transmission but also adapt to the dynamic nature of the IoT network in such environments. By developing a trusted path selection strategy that

accounts for link stability, energy levels, and trust relationships, we can achieve a dependable and efficient routing mechanism for IoT networks in edge computing scenarios [20][21]. This will not only enhance the overall network performance but also safeguard against potential security threats, providing a robust foundation for various IoT applications in real-world settings.

In this paper, we propose a novel Stability-based On-demand Multi-path Distance Vector (STAOMDV) algorithm that addresses the unique challenges of edge computing scenarios in IoT networks. The STAOMDV algorithm comprehensively considers the link life cycle, capability level, and trust level to improve network performance while ensuring security. By integrating the proposed trusted path selection strategy with the STAOMDV algorithm, we establish a dependable and efficient routing mechanism for IoT networks in edge computing scenarios. Our contribution is to provide an innovative approach that addresses the limitations of existing routing protocols, facilitates secure data transmission, and improves the overall reliability and efficiency of IoT networks in edge computing environments.

The organizational structure of the paper is as follows: Section 2 presents an overview of the system's model, focusing on the network model and the utilization of a directed weighted graph. Section 3 introduces the trusted path selection policy, which considers link life cycle, energy level, trust level, and authentication status to optimize data forwarding quality and security in edge computing scenarios. In Section 4, the design and implementation of the STAOMDV routing protocol are explicated, outlining how it achieves stability-based multi-path routing in edge environments. Section 5 elucidates the results of our experimental analysis, providing empirical evidence of the proposed model's effectiveness and benefits. Finally, Section 6 encapsulates the main findings of this study in the form of a comprehensive summary and outlines potential future research directions in this field.

2. System Model

A trusted routing mechanism for edge computing scenarios is proposed, and the network model involved will be detailed in this section. The network model is abstracted as a directed weighted graph for better description. This abstraction allows for a more precise depiction of the network's topology, enabling a rigorous examination of the proposed mechanism's effectiveness and performance in the context of edge computing scenarios.

2.1 Network model

A multi-trust domain scenario is considered, where each trust domain contains multiple mobile Internet of Things (IoT) devices and an edge server infrastructure called Mobile Edge Computing (MEC), such as base stations (BS), gateways, and roadside units (RSU). IoT devices within the same trust domain form a mobile ad hoc network through self-organization [22]. In the mobile ad hoc network, device collaboration relies on data exchange, and two devices that are not within each other's communication range need to rely on other intermediate nodes to forward data transmission. Data transmission is achieved by establishing multi-hop routing between the source node and the destination node. However, due to the large scale of the IoT network, too many forwarding hops may cause an unstable transmission path for distant destination nodes. Therefore, in this paper, routing planning is achieved through the partitioning of trust domains. As shown in **Fig. 1**, the communication modes include Device-to-Device (D2D) communication, Device-to-MEC (D2M) communication, and MEC-to-MEC (M2M) communication.

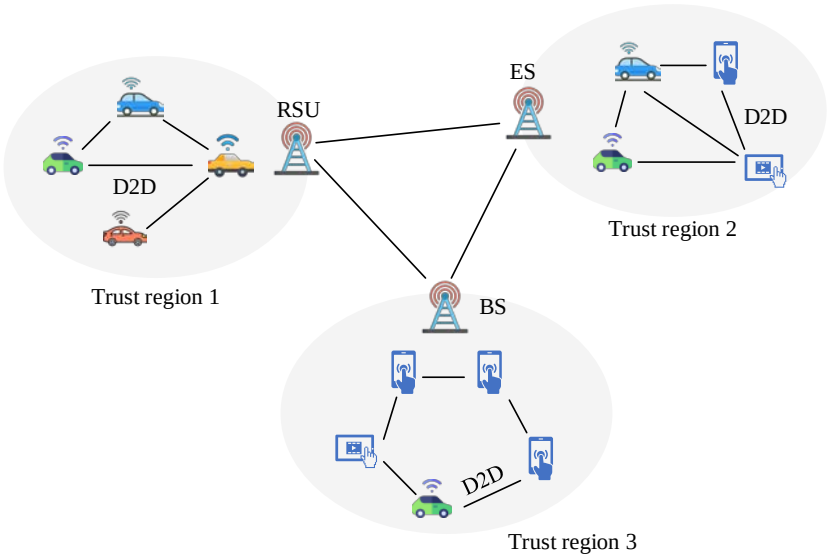


Fig. 1. Network model

2.2 Directed weighted graph

In order to facilitate the description and analysis, the network model is abstracted as a directed weighted graph:

$$G = \{N, E, W\} \tag{1}$$

where, $N = \{n_1, n_2, ..., n_N\}$ represents the mobile device in the network, $E = \{e_{ij} \mid i, j \in N\}$ represents the edge set and W represents the set of weights for edges. Specifically, e_{ij} represents the connection relationship between nodes n_i and n_j. If the two are within the communication range of each other, then $e_{ij} = e_{ji} = 1$. The weights of edge e_{ij} consist of a set of attribute values represented as $< L(n_i, n_j), E(n_i, n_j), T(n_i, n_j), Auth(n_i, n_j) >$. Among these attributes, $L(n_i, n_j)$ represents the lifecycle, $E(n_i, n_j)$ represents the energy level, $T(n_i, n_j)$ represents the trust level, and $Auth(n_i, n_j)$ represents the authentication status. The specific meanings of these attributes will be explained in the next section. We use $\mathbf{P_{s \to d}} = \{P_1, P_2, ...\}$ to represent the set of paths from a source node to a destination node, and $P_s =< n_s, n_s^1, ..., n_s^m, n_d >, P_s \in \mathbf{P_{s \to d}}$ represents the nodes traversed in sequence P_s.

3. Trusted Path Selection Policy

The traditional AOMDV routing protocol may not be suitable for edge computing scenarios as IoT devices are often resource-constrained and operate in dynamic and unpredictable environments. Therefore, a trusted path selection strategy is necessary to improve the quality and security of data forwarding. The proposed strategy selects the best route based on a combination of link life cycle, energy level, and trust level and authentication status. Link life cycle refers to the expected operational duration of a link, based on its historical behavior. Energy level refers to the remaining battery life of a node, an important consideration for IoT devices with limited energy resources. Trust level refers to a node's reputation in the network,

evaluated based on historical behavior and interactions with other nodes. By considering all three factors, the proposed strategy enables the selection of an optimal route that not only optimizes network performance but also ensures security by avoiding potential disruptions caused by malicious nodes during data forwarding.

3.1 Link life cycle

In edge computing scenarios, the mobility of IoT devices can lead to the disconnection of link connections between adjacent devices as they move. Therefore, the consideration of link lifecycle becomes crucial when selecting routes. The link lifecycle is defined as the duration of the connection between two nodes and is estimated based on the relative speed of the two nodes. Due to the mobility of IoT devices, the link connection between adjacent devices may break, making it essential to factor in the link lifecycle when routes are being chosen. Specifically, for the link lifecycle between adjacent nodes, it can be estimated based on the relative speed of the two nodes.

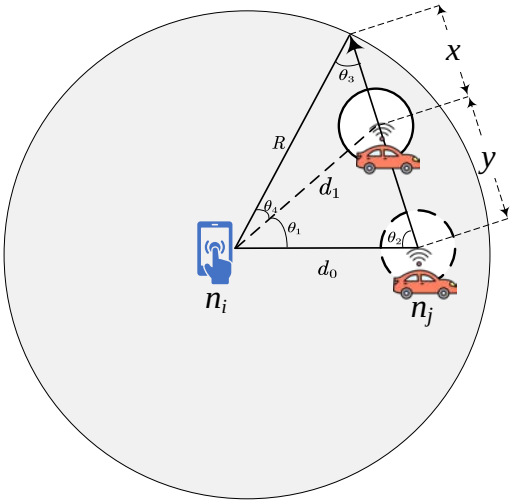


Fig. 2. Illustration of link time between neighboring nodes

As shown in **Fig. 2**, assuming that at time t_0, node n_j is at point a and its distance from node n_i is d_0, and at time t_1, node n_j is at point b and its distance from node n_i is d_1. Node n_j reaches the maximum communication distance R with node n_i at point c. The values of d_0 and d_1 can be estimated based on the received signal strength indicator (*RSSI*) [23], and the angle is θ_1. Using the estimated values of d_0 and d_1 based on *RSSI*, as well as θ_1, we can calculate the link lifetime L_{ij} between nodes n_i and n_j.

According to the cosine law, the relative displacement of node n_j 's movement within time interval Δt can be calculated as:

$$y = \sqrt{d_0^2 + d_1^2 - 2d_0 d_1 \cos\theta_1} \tag{2}$$

According to the sine theorem, it can be obtained that:

$$\frac{d_1}{\sin\theta_2} = \frac{y}{\sin\theta_1}, \frac{d_0}{\sin\theta_3} = \frac{R}{\sin\theta_2} \tag{3}$$

Then it can be solved sequentially to get θ_2, θ_3, then we can get $\theta_4 = 180° - \theta_2 - \theta_3$. The distance between b of n_j and the point c beyond the communication distance of n_i the node can be calculated:

$$x = \sqrt{R^2 + d_1^2 - 2Rd_1\cos\theta_4} \tag{4}$$

The moving speed of n_j can be calculated as following:

$$v_j = \frac{y}{\Delta t} \tag{5}$$

where $\Delta t = t_1 - t_0$.

From the above formula, the life cycle of link $n_i \to n_j$ can be estimated as:

$$L\left(n_i, n_j\right) = \frac{x}{v_j} \tag{6}$$

For the path P_s, there are multiple nodes, then its link life cycle is determined by the link with the shortest life cycle in P_s, described as:

$$L_{P_s} = \min\left\{L\left(n_s, n_s^1\right), L\left(n_s^1, n_s^2\right), ..., L\left(n_s^m, n_d\right)\right\} \tag{7}$$

3.2 Link energy level

Limited by the limitations of the device itself, the link between nodes will also be disconnected when the device energy is too low. Therefore, choosing a path with higher energy level can effectively ensure the stability of the link. We use the remaining energy ratio to describe the energy level of the device. The remaining energy ratio of node a is defined as:

$$RER_j = \frac{E_j^{residual}}{E_j^{initial}} \tag{8}$$

where $E_j^{residual}$ represents the remaining energy of n_j, while $E_j^{initial}$ represents the initial energy of n_j.

The link energy level of link $n_i \to n_j$ depends on the energy level of the next hop node:

$$E\left(n_i, n_j\right) = RER_j \tag{9}$$

For a path P_s with multiple hop nodes, the energy level of the link is determined by the node with the lowest energy level among all the intermediate nodes it passes through. From this, the energy level of link P_s can be obtained as:

$$E_{P_s} = \min\left\{RER_{n_s^1}, RER_{n_s^2}, ..., RER_{n_s^m}\right\} \tag{10}$$

3.3 Link trust level

Due to the existence of malicious or selfish nodes in the IoT environment, the data forwarding process may be subject to malicious attacks. Therefore, selecting a trusted data forwarding path is crucial for the security of routing. In this paper, we select a high-trust path based on the trust value of nodes. The trust calculation method has been implemented in our previous work [24], mainly through the assistance of edge servers to measure the trust between nodes. Specifically, we select trust attributes related to data forwarding to measure the trust of forwarding nodes, mainly including data packet successful forwarding rate, data packet

duplication rate, and communication delay.

Data packet successful forwarding rate refers to the ratio of the number of correctly forwarded packets to the number of packets that should have been forwarded [25]. The trust component corresponding to the data packet successful forwarding rate is calculated as:

$$dt_{FCR}\left(n_i,n_j\right)=FCR\left(n_i,n_j\right)=\frac{FPN_{ij}\left(\Delta\tau\right)}{RPN_{ij}\left(\Delta\tau\right)} \tag{11}$$

where, $FPN_{ij}(\Delta\tau)$ represents the number of packets that n_j successfully forwarded within a certain time interval $\Delta\tau$, and $RPN_{ij}(\Delta\tau)$ represents the number of packets that n_j was requested to forward within a certain time interval $\Delta\tau$.

The data packet duplication rate refers to the proportion of forwarded duplicate packets. This rate is used to effectively identify the attack and can be calculated as follows:

$$PRR\left(n_i,n_j\right)=\frac{PRN_{ij}\left(\Delta\tau\right)}{PN_{ij}\left(\Delta\tau\right)} \tag{12}$$

where, $PRN_{ij}(\Delta\tau)$ represents the number of forwarded packets that are duplicates within a certain time interval $\Delta\tau$ by node n_j, and $PN_{ij}(\Delta\tau)$ represents the number of packets that n_j was requested to forward within a certain time interval $\Delta\tau$.

The trust metric calculated through data packet duplication rate calculation is shown as:

$$dt_{PRR}\left(n_i,n_j\right)=\begin{cases}2-\alpha^{b} & b<\delta\\ 0 & otherwise\end{cases} \tag{13}$$

where, $\alpha>1$, $b=PRR(n_i,n_j)$, $\alpha^{\delta}=2$. The closer the data packet duplication rate of a node is to the threshold, the more likely it is that the node is a malicious node.

Communication delay refers to the communication delay of data transmission between nodes, which should fluctuate within a normal range. In particular, for delay-sensitive services, if the communication delay is too long, the likelihood of nodes launching malicious behavior increases [26]. Assuming the delay when n_j forwards the data packet to n_i is $Delay(n_i,n_j)$, the calculation of the trust component corresponding to the communication delay is shown as:

$$dt_{Delay}\left(n_i,n_j\right)=\begin{cases}\beta^{\frac{d-\sigma}{\sigma}} & d\ldots\sigma\\ 1 & otherwise\end{cases} \tag{14}$$

where, $\beta=0.01$, $d=Delay(n_i,n_j)$, the threshold σ is dependent on the specific message.

Combining the three trust attributes related to communication behavior mentioned above, n_i can obtain the overall trust $Trust_{ij}$ of n_j. From this, the trust level of Link $n_i\rightarrow n_j$ can be obtained as:

$$T\left(n_i,n_j\right)=Trust_{ij} \tag{15}$$

For path P_s, which involves multiple nodes, the trust level is determined by the link with the lowest trust level. From this, the trust level of Link P_s can be obtained as:

$$T_{P_s}=\min\left\{T\left(n_s,n_s^1\right),T\left(n_s^1,n_s^2\right),...,T\left(n_s^m,n_d\right)\right\} \tag{16}$$

3.4 Link authentication status

In an open IoT environment, unauthenticated and unauthorized nodes can threaten the security

and privacy of nodes. To ensure high security for data transmission, it is essential to select an authenticated link for the data transmission process. If adjacent nodes n_i and n_j implement mutual authentication, the identity authentication status of Link e_{ij} and e_{ji} is 1, i.e., $Auth(n_i, n_j) = Auth(n_j, n_i) = 1$, otherwise it is 0.

For path P_s, if all the nodes it passes through and the next-hop node implement mutual authentication, then the path authentication status is 1, i.e., $Auth_{P_s} = 1$, otherwise $Auth_{P_s} = 0$.

By controlling the authentication status of nodes along the data forwarding path, it becomes feasible to effectively prevent unauthorized malicious nodes from discarding or tampering with data packets. This ensures the seamless transmission of data that demands high security. Conversely, for data with lower security requirements, where security is not a major concern, priority can be given to efficiency. In such cases, it is permissible to allow data forwarding by nodes that have not undergone identity authentication.

3.5 Trusted path selection strategy

The link's lifecycle, energy level, and trust level can all describe the stability of the link from different perspectives. This section will integrate the three factors of link stability to implement a trusted path selection strategy. Assuming that the set of alternative paths from source node n_s to destination node n_d is $\mathbf{P_{s\to d}} = \{P_1, P_2, ..., P_k\}$, the link stability of the alternative path $P_s \in \mathbf{P_s}$ is calculated as:

$$STA(P_s) = w_1 L_{P_s} + w_2 E_{P_s} + w_3 T_{P_s} \tag{17}$$

where, w_1, w_2, w_3 represent the weights of the link's lifecycle, energy level, and trust level, respectively.

Obviously, the trusted path selection based on link stability is a multi-attribute decision-making process. In the decision-making process, if an attribute only produces small differences among all alternative options, it means that the impact of this attribute on the decision-making process is small. In other words, the greater the difference in the attribute values among all alternative options, the greater the influence of this attribute on the decision-making process. Objective information entropy can be employed to measure the extent of attribute dispersion, enabling the determination of weights for the mentioned attributes. The attribute matrix is formed by the decision attributes of all paths in the set of alternative paths, which is shown as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ ... & ... & ... \\ a_{k1} & a_{k2} & a_{k3} \end{bmatrix} \tag{18}$$

where, $a_{k1} = L_{P_k}$, $a_{k2} = E_{P_k}$ and $a_{k3} = T_{P_k}$.

First, the attribute matrix is normalized to obtain a normalized matrix A', which is denoted as A for convenience. Then, objective information entropy is used to calculate the weights. The information entropy of attribute i is calculated as:

$$E_j = -\ln(k) \sum_{i=1}^{k} p_{ij} \ln p_{ij},\ j = 1, 2, 3 \tag{19}$$

$$p_{ij} = \begin{cases} a_{ij} / \sum_{i=1}^{k} a_{ij} & \sum_{i=1}^{k} a_{ij} \neq 0 \\ 0 & otherwise \end{cases} \tag{20}$$

Then, the weights of each attribute are calculated as followed:

$$w_i = \left(1 - E_i\right) \Big/ \left(3 - \sum_{j=1}^{3} E_j\right) \tag{21}$$

Therefore, based on equations 17 - 21, the link stability of all paths in the set of alternative paths can be calculated, and the optimal path can be selected as the data transmission path, as shown in **Algorithm 1**. In the scenario considered in this paper, the source node and destination node may be located in different trust domains, as shown in **Fig. 3**. In this case, the best communication mode and path are selected by evaluating the stability of links under different communication modes, where the red path represents the hybrid mode, which is achieved through the collaboration of edge servers for data forwarding, including D2D, D2M, and M2M communication modes; and the green path only includes the D2D mode. In the path selection process, all paths in the routing table with the destination nodes of both n_d and MEC are taken as the set of alternative paths, and the link stability of all paths in the set is evaluated, and the highest one is selected as the data transmission path.

Algorithm 1 Trusted Path Selection Algorithm

Input: Source node n_s, Destination node n_d, Candidate path set to $n_d\mathbf{P_{s\to d}}$, Candidate path set to MEC

Output: Best transmission path P_s

if $e_{sd} = 1$ **then**
  Directly transmit data from n_s to n_d, i.e., $P_s = \{n_s, n_d\}$;
  Return P_s;
end if
if n_s and n_d are in the same trust domain **then**
  Initialize the candidate path set $\mathbf{P_s} = \mathbf{P_{s\to d}}$;
  Build the decision attribute matrix A based on the candidate path set $\mathbf{P_s}$;
  Calculate the attribute weights w_1, w_2, w_3 according to Equations 19-21;
  Calculate the link stability $\{STA_{P_{s'}} \mid P_{s'} \in \mathbf{P_s}\}$ of all candidate paths according to Equation 17;
  $P_s = arg\max\{STA_{P_{s'}} \mid P_{s'} \in \mathbf{P_s}\}$;
else
  $\mathbf{P_s} = \mathbf{P_{s\to d}} \cup \mathbf{P_{s\to MEC}}$;
  Select the best path $P_s = arg\max\{STA_{P_{s'}} \mid P_{s'} \in \mathbf{P_s}\}$ according to Steps 7-10; %If $P_s \in \mathbf{P_{s\to d}}$, select the D2D mode for data transmission. If $P_s \in \mathbf{P_{s\to MEC}}$, select the hybrid mode for data transmission;
end if
return P_s;

In the above link-stability-based path selection, if there are multiple paths with maximum link stability, more than one optimal path is selected, then the path with the least number of hops is chosen as the final data transmission path. If there are still multiple paths selected based on path hops and link stability, then one of them is randomly selected.

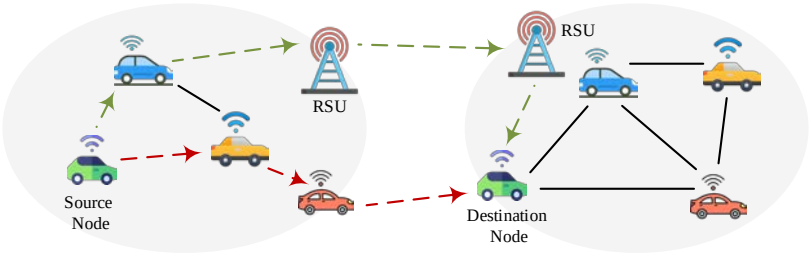


Fig. 3. Edge-cooperative-based data forwarding path selection

4. STAOMDV: Stability-based On-demand Multi-path Distance Vector

The STAOMDV routing algorithm involves the discovery of multiple paths from the source node to the destination node, followed by the collection of relevant decision attributes, and ultimately selecting trustworthy routes based on link stability. This section presents a comprehensive depiction of the workflow of the STAOMDV routing protocol. It begins with an overview of the overall routing process of STAOMDV, followed by detailed descriptions of the routing information table design, the routing discovery process, and the routing maintenance process.

4.1 STAOMDV routing process

Algorithm 2 STAOMDV routing algorithm

Input: source node n_s ;destination node n_d ;authentication status requirement *Tag*
Output: best transmission path P_s

Query the routing table to obtain the set of candidate routes that meet the authentication status requirements $\mathbf{P_{s\to d}}$;
if $\mathbf{P_{s\to d}} = NULL$ **then**
  Generate the RREQ message and broadcast it;
  if NET_TRAVERSAL_TIME **then**
    Receive the RREP message;
    Construct the corresponding routing table entry;
  else
    return;
  end if
end if
if n_s and n_d are in the same trust domain **then**
  Query the routing table for the set of alternative route $\mathbf{P_{s\to d}}$;
else
  Repeat step 1-10 with MEC as the destination node;
  Construct the candidate paths set $\mathbf{P_{s\to MEC}}$;
end if
Get the best transmission path P_s according to **Algorithm 1;** % n_s sends data according to P_s and listens to neighboring nodes for forwarding, then updates the trust and maintains the route.
return P_s ;

As depicted in **Algorithm 2**, the STAOMDV routing process primarily consists of the following steps:

Step 1: When the source node n_s has data to send to the destination node n_d, it first checks whether there is a routing table entry in the routing table that meets the authentication requirements and can reach the destination node n_d. If such an entry exists, the source node n_s constructs the candidate paths set $\mathbf{P}_{\mathbf{s}\to\mathbf{d}}$; otherwise, it proceeds to execute Step 2.

Step 2: The source node n_s generates a route request message RREQ as required and broadcasts it. If it receives a route reply message RREP within the valid time, it constructs the candidate paths set $\mathbf{P}_{\mathbf{s}\to\mathbf{d}}$; otherwise, the program ends.

Step 3: n_s determines whether it is in the same trust domain as n_d, if not, find the route or perform route discovery process with MEC as the destination node and get the candidate paths set $\mathbf{P}_{\mathbf{s}\to\mathbf{MEC}}$.

Step 4: Call **Algorithm 1** to obtain the optimal trusted path P_s.

Step 5: n_s sends data according to the obtained optimal path and listens to the neighboring nodes, then updates the trust value to the neighboring nodes and maintains the routes.

The above routing discovery and routing maintenance process will be described in detail later, and the trust value of nodes is regularly maintained to detect malicious neighboring nodes in a timely manner.

4.2 Routing information table

Each node in the STAOMDV routing protocol needs to maintain a routing information table to store its routing information to other nodes. As shown in **Fig. 4**, compared to the standard AOMDV routing protocol, the STAOMDV routing table has four additional fields $L_P, E_P, T_P, Auth_P$ to store the three link stability attributes and link authentication status mentioned in the previous section.

Destination IP
Sequence Number
Advertised Hop Count
Expiration Timeout
Route List {(NextHop1, HopCount1, LastHop1,Timeout1, $L_{P_1}, E_{P_1}, T_{P_1}, Auth_{P_1}$), (NextHop2, HopCount2, LastHop2,Timeout2, $L_{P_2}, E_{P_2}, T_{P_2}, Auth_{P_2}$), … }

Fig. 4. Routing table structure of STAOMDV

Compared to the routing information table in the standard AOMDV routing protocol, four new fields are added, which increases the storage consumption of the nodes. Considering the limited storage resources of the nodes, we consider the integer data between to represent the three fields to reduce the storage consumption of the nodes.

4.3 Routing discovery process

AOMDV routing protocol is an on-demand multipath routing protocol. When a node needs to transmit data, it needs to select a trusted path from the set of alternative paths for data

transmission. If the set of alternative paths is empty or none of them meet its requirements, the node needs to perform route discovery process to discover multiple paths to the destination node. The route discovery process is accomplished by two control messages, Route Request (RREQ) and Route Reply (RREP) [27][28]. The transmission of the two messages, RREQ and RREP, is used to establish routing relationships between nodes and to collect path stability properties.

4.3.1 Routing request

As shown in **Fig. 5**, the routing request message RREQ in STAOMDV contains the following fields: (1) *RREQ ID*, which indicates the identification number of the route request message; (2) *Destination IP Address*, which indicates the IP address of the destination node; (3) *Destination Sequence Number*, which indicates the destination node sequence number; (4) *Source IP Address*, which indicates the IP address of the source node; (5) *Source Sequence Number*, which indicates the source node sequence number; (6) *First Hop*, which indicates the first forwarding node through which the route request message passed; (7) $ReverseL_P$, represents the life cycle of the reverse path; (8)$ReverseE_P$, represents the energy level of the reverse path; (9) $ReverseT_P$, represents the trust level of the reverse path; (10) $ReverseAuth_P$, represents the authentication status of the reverse path; (11) *Tag*, represents the authentication status required by the source node. The last four fields are unique to STAOMDV to store the link stability properties and authentication status of the reverse path, which is continuously updated by RREQ during propagation to establish a path for the node in the opposite direction of RREQ propagation.

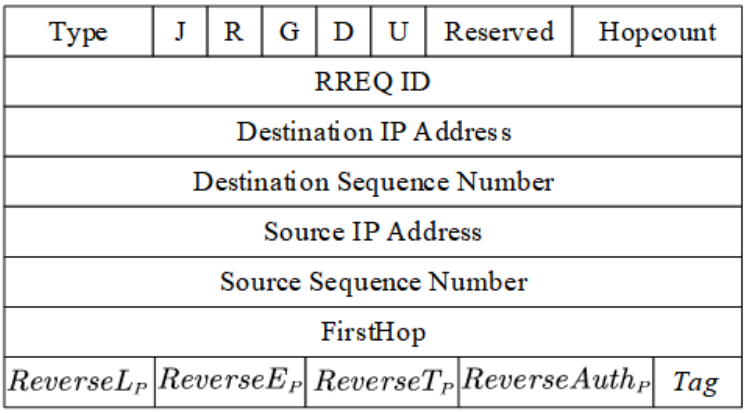


Fig. 5. RREQ packet structure of STAOMDV routing protocol

When the source node constructs a route request message, it sets the fields *FirstHop* to NULL, $ReverseL_P$, $ReverseE_P$, $ReverseT_P$ and $ReverseAuth_P$ to NULL, 1, 1, and NULL, respectively. Intermediate nodes in different states handle the RREQ message differently when they receive it. Suppose an intermediate node n_u receives an RREQ message broadcasted by a neighbor node n_v , its processing is as follows:

Step 1: Determine whether the remaining energy of itself is at the normal level, whether the trust value of the neighbor node is within the security range, whether the hop count of the current RREQ message is within the maximum hop count and whether the authentication status is consistent with the source node requirement, i.e., determine whether $RER_u > TH_{ENE}$ && $Trust_{uv} > TH_{TR}$ && $Hopcount$,, TH_{hop} is satisfied. If it is satisfied, Step 2 is executed; otherwise, the RREQ message is discarded.

Step 2: Check whether a routing table entry to the node n_v exists in the routing table. If not, create a route to the node n_v in the routing table and set the value of the field storing the link stability attribute as $L_P = L(n_u, n_v)$, $E_P = 1$, $T_P = T(n_u, n_v)$ and $Auth_P = Auth(n_u, n_v)$.

Step 3: Check if the same RREQ message is received. If yes, determine whether the reverse route constructed by the current RREQ message copy intersects with the reverse route link constructed by the previous RREQ message copy, and if so, discard the RREQ message copy and end the procedure; otherwise, execute Step 4.

Step 4: If the reverse route constructed from the later received copy of the RREQ message is larger than all the reverse route hops to the source node existing in the current routing table and the calculated link stability is lower, discard the copy of the RREQ message and terminate the procedure; otherwise, execute Step 5.

Step 5: If n_u is not the source node, create a reverse route with the source node as the destination node and set the routing table fields as $L_P = \min\{ReverseL_P, L(n_u, n_v)\}$, $T_P = \min\{ReverseT_P, T(n_u, n_v)\}$, and authentication status is set according to the definition.

Step 6: If a route with n_d as the destination node exists in n_u 's routing table, and satisfies *Sequence Number* > *Destination Sequence Number* and meets the required authentication status, n_u generates a route answer message and unicasts it to n_v; otherwise, Step 7 is executed.

Step 7: n_u first determines whether it is the first hop node, and if so, sets $FirstHop = n_u$, $ReverseL_P = L(n_u, n_v)$, $ReverseAuth_P = Auth(n_u, n_v)$. Then update the other fields of RREQ as $HopCount = HopCount + 1$, $ReverseL_P = \min(ReverseL_P, L(n_u, n_v))$, $ReverseE_P = \min(ReverseE_P, ENE_u)$, $ReverseT_P = \min(ReverseT_P, T(n_u, n_v))$ and set $ReverseAuth_P$ according to the definition. Finally, n_u broadcasts the updated RREQ message to the neighboring nodes.

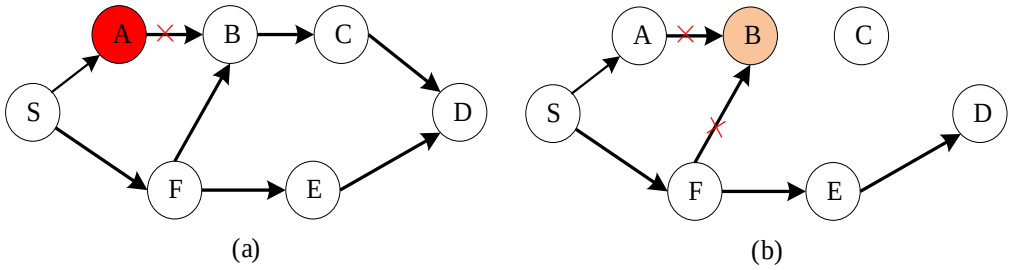


Fig. 6. Discard handling in RREQ message transmission:
(a) malicious node exists; (b) node energy level is low

In the processing of the above route request message RREQ, by judging whether the previous hop node is a malicious node before processing, such as node A in **Fig. 6(a)**, the propagation of malicious messages made by malicious nodes is blocked in time to reduce unnecessary network overhead; by judging its own energy level to choose whether to continue to forward RREQ messages, such as node B in **Fig. 6(b)**, when the energy level is too low, it does not forward the RREQ messages so that it does not participate in the forwarding of other new routes and only participates in the data forwarding of existing routes. In addition, the maximum number of hops for node route establishment is controlled by the hop count

threshold TH_{hop}, so that route establishment is restricted to nodes within the same domain or adjacent domains to avoid long paths, which can usually be determined by the optimal number of hops between the two nodes with the farthest distance in the trust domain.

4.3.2 Routing reply

As shown in **Fig. 7**, the routing reply message RREP in STAOMDV contains the following fields: (1) *Destination IP Address*, which represents the IP address of the destination node; (2) *Destination Sequence Number*, which represents the serial number of the destination node; (3) *Source IP Address*, which represents the IP address of the source node; (4) *Source Sequence Number*, which represents the serial number of the source node; (5) *First Hop*, which represents the first forwarding node through which this routed reply message passes; (6) *LifeTime*, which represents the valid time of the RREP and is valid only for messages received within the valid time; (7) $ForwardL_P$, which represents the life cycle of the forward path; (8) $ForwardE_P$, which represents the energy level of the forward path; (9) $ForwardT_P$, which represents the trust level of the forward path; (10) $ForwardAuth_P$, which represents the authentication status of the forward path. Similar to the RREQ message, the last four fields are used to store the link stability attributes and path authentication status of the forward path, respectively.

Type	R	A	Reserved	Hopcount
Destination IP Address				
Destination Sequence Number				
Source IP Address				
Source Sequence Number				
FirstHop				
LifeTime				
$ForwardL_P$	$ForwardE_P$	$ForwardT_P$	$ForwardAuth_P$	

Fig. 7. RREP packet structure of STAOMDV routing protocol

There are two situations in which a route reply message (RREP) is generated:

(1) When there is a route to the destination node n_d in the routing table of the intermediate node n_u, and the sequence number of this routing table entry is newer than that of RREQ, the RREP message is generated. The *FirstHop*, *ForwardL*$_P$, *ForwardE*$_P$ and *ForwardT*$_P$ fields in the RREP are respectively set to the *LastHop*, L_P, $\min\{ENE_u, E_P\}$ and T_P from the queried routing table entry. It is then unicasted to node n_v.

(2) When the destination node n_d receives the RREQ message from a neighbor, it generates the route reply message RREP. If the Destination Sequence Number is equal to the sequence number maintained by itself, the maintained sequence number is incremented by one; if they are equal, the maintained sequence number remains unchanged. Field settings are similar to when generating the RREQ message. The RREP is then unicasted to the neighbor node that sent the RREQ, and n_d only replies to the first RREQ sent by the same neighboring node.

When an intermediate node n_u receives an RREP from node n_v, its processing procedure is as follows:

Step 1: Node n_u checks its routing table for a route to node n_v. If no route exists, it creates a route to n_v in the routing table, setting the field values for link stability attributes as $L_P = L(n_u, n_v)$, $E_P = 1$, $T_P = T(n_u, n_v)$ and $Auth_P = Auth(n_u, n_v)$.

Step 2: Node n_u examines its routing table for a route to node n_d. If none is found, it creates a forward route with n_d as the destination node, setting the fields in the routing table to $NextHop = n_v$, $LastHop = FirstHop$, $E_P = ForwardE_P$, $L_P = \min\{ForwardL_P, L(n_u, n_v)\}$, $T_P = \min\{ForwardT_P, T(n_u, n_v)\}$ and the path authentication status.

Step 3: Node n_u updates the fields in the RREP, a process similar to what's done with RREQ. It then checks if there exists in the routing table a reverse route that has not been used to reply with an RREP. If there is one that meets the authentication status requirements, it's selected for sending the RREP; otherwise, the RREP is discarded.

When the source node n_u receives the RREP message, it establishes the corresponding routing table entry to the destination node n_d based on the fields within the RREP. At the end of route discovery, node n_s selects the optimal route for data transmission from the established multiple paths and the collected link stability attributes, using a trusted path selection mechanism.

4.4 Routing maintenance

The state of IoT networks in edge computing scenarios is dynamically changing, and the routing information stored by nodes may become invalid due to node movement, energy consumption, or trust value changes. Timely maintenance of routing table information of nodes in the network can effectively reduce the frequency of route discovery, and can reduce the path selection failure or not optimal due to outdated link information. Based on the AOMDV routing protocol, STAOMDV routing implements link stability property updates and error route feedback.

4.4.1 Link stability property update

When the stability of a single-hop link in the network, i.e., the edge weights in the directed weighted graph, varies too much, it can lead to the inability of nodes to accurately assess the stability of the path when performing path selection. Therefore, an event-driven mechanism for updating link stability attributes is proposed in this paper. There are three trigger conditions for this mechanism, as shown in the link e_{FE} in **Fig. 8**. The mechanism is triggered when the life cycle, energy level or trust level changes too much, as follows:

$$\begin{aligned} \Delta L_{FE} &\geq TH_L \\ \Delta E_{FE} &\geq TH_E \\ \Delta T_{FE} &\geq TH_T \end{aligned} \tag{22}$$

where, ΔL_{FE}, ΔE_{FE}, and ΔT_{FE} are the change rates of life cycle, energy level, and trust value obtained from the difference in monitoring values of the three indicators measured by the node twice consecutively; TH_L, TH_E, and TH_T denote the change rate thresholds of the three attributes, used to control the maximum degree of acceptable change.

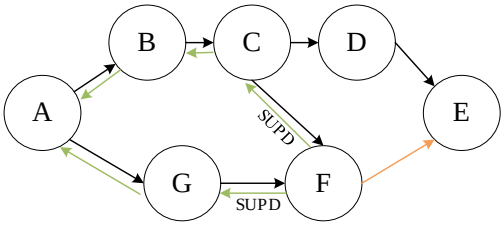


Fig. 8. Link stability attribute update trigger indication

When any one of the above three trigger conditions is sensed, node F updates the attribute field values of all route entries with E as the next hop node and generates a stability attribute update packet (SUPD) that is broadcast to neighboring nodes. For example, the orange arrow in **Fig. 8** indicates that at least one of the above events occurs, and the green arrow indicates the SUPD flow of the attribute update message.

The structure of the SUPD message is shown in **Fig. 9**, where *Source IP Address* indicates the source node of the route to update the stability attributes; *Source Sequence Number* indicates the serial number of the source node; *Update Destination List* indicates the set of routes to update the stability attributes, including the destination node address and last hop node address of each route, and the destination node serial number; $UpdateL_P$, $UpdateE_P$, and $UpdateT_P$ indicate the perceived changed link life cycle, energy level, and trust level, respectively.

Type	Reserved	Updatecount
Broadcast ID		
Source IP Address		
Source Sequence Number		
Update Destination List		
$UpdateL_P$	$UpdateE_P$	$UpdateT_P$

Fig. 9. SUPD packet structure of STAOMDV routing protocol

4.4.2 Routing errors

Single-hop links may fail due to increased node distance, energy depletion, or being identified as a malicious node. In standard AOMDV, the failed routes in the network are removed by Route ERROR (RERR) messages. The execution of RERR messages in STAOMDV is similar to that of SUDP. Unlike SUDP, the link is actively disconnected when a node finds a neighbor node with a low trust value and considers it as a malicious node, or the node sends a RERR message when the link is disconnected due to a low neighbor node energy and distance beyond the communication distance (which can be obtained through link layer feedback). Similarly, the node handles the RERR message in a similar way as SUDP after receiving it. As shown in **Fig. 10**, node F finds that the link to E is unreachable or finds that node E is a malicious node, it constructs and broadcasts a RERR message, whose flow direction is shown by the red arrow, and the node receiving the RERR message deletes the relevant routing table entries.

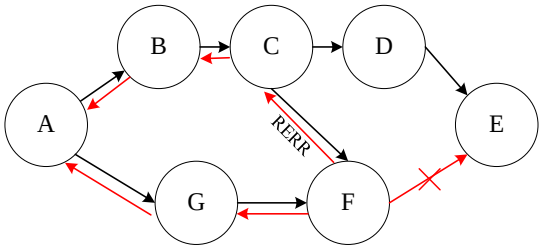


Fig. 10. RERR example in STAOMDV

5. Experimental results and analysis

To verify the effectiveness of the proposed trusted routing mechanism STAOMDV, simulation experiments were designed and conducted. The analysis involved comparing the performance of STAOMDV with other routing protocols. The experimental process encompassed setting the relevant parameters, selecting appropriate evaluation metrics, and presenting the obtained results. The experiments were conducted on a PC machine with Intel(R) Core(TM) i5-9400F CPU operating at 2.90 GHz and 16GB of memory.

5.1 Experimental setting

The NS-3 platform was utilized to implement simulation experiments for the STAOMDV routing protocol. The experimental scenario consisted of a single area $1000m \times 1000m$, where 50 mobile nodes were randomly deployed. The communication range was set to a predetermined value, and the mobility of nodes was simulated through a random movement model. To evaluate the effectiveness of the STAOMDV protocol, we compared its performance with two other routing protocols: AOMDV and LLECP-AOMDV [29]. The simulation parameters used in the experiments were established according to **Table 1**. Besides the conventional terminal device parameters, specific protocol parameters relevant to this paper were also defined. To replicate the movement status of IoT devices, a random movement model was employed, and different movement speeds were set to closely resemble real-world IoT scenarios. Additionally, the participation of forwarding devices was regulated by energy thresholds, where devices with a remaining energy rate below 0.2 were deemed insufficient for data forwarding activities. Routing information maintenance was triggered by the rate of change threshold, signifying the need for maintenance when the change exceeded 0.1.

Table 1. Parameters setting of simulation experiment

Parameters	Values
Simulation time	600s
Type of communication	CBR
Type of MAC	802.11DCF
Mobile model	Random movement model
Packet size	512bytes
Packet transmission rate	2M bits/s
Queue buffering	50
Initial energy	50J
Number of connections	20
Energy threshold TH_{ENE}	0.2
Trust value threshold	0.4
TH_L, TH_E, TH_T	0.1, 0.1, 0.1

In order to accurately measure and evaluate the experimental results, the end-to-end average delay, group delivery rate and the number of energy-depleted nodes are selected as evaluation metrics to analyze and compare the experimental results.

(1) AED (average end-to-end delay). It represents the average time it takes for a packet to be transmitted from the source node to the destination node and is calculated as follows:

$$AED = \frac{1}{n}\sum_{i=1}^{n}\left(t_i^{arrive} - t_i^{sent}\right), \tag{23}$$

where, n is the number of successfully delivered data packets, t_i^{sent} is the time when data packet i was sent, and t_i^{arrive} is the time when data packet i arrived at the destination node.

(2) PDR (packet delivery rate). It represents the ratio of the number of data packets received by the destination node to the number of data packets sent out by the source node. The higher the packet delivery rate is, the better the protocol effect is. It is calculated as follows:

$$PDR = \frac{Num_{arrive}}{Num_{sent}} \times 100\%, \tag{24}$$

where, Num_{arrive} represents the number of data packets arriving at the destination node and Num_{sent} represents the number of data packets sent by the source node.

(3) NEN (number of energy exhausted nodes). It is the sum of the number of nodes whose energy is zero at the end of the simulation and is calculated as follows:

$$NEN = \left|\left\{n \middle| n \in N \wedge E_n = 0\right\}\right| \tag{25}$$

where, n is the node of the network, E_n is the remaining node energy. || represents the number of elements in the set.

5.2 Experimental results analysis

The proposed trusted routing protocol in this paper integrates the path lifecycle, energy level, and trust level to enhance its efficacy. To validate its effectiveness, we conducted simulation experiments to examine the network performance under varying maximum device mobility rates and different numbers of malicious nodes. The evaluation of STAMODV's performance focuses on achieving balanced energy consumption by monitoring the number of energy-depleted nodes in the network.

5.2.1 Effect of node movement speed

This part of the simulation experiment assumes the presence of four malicious nodes in the network to simulate malicious attacks (e.g., black hole attack, gray hole attack) during the routing process. In this paper, the malicious behavior is assumed to be black hole attack. By varying the maximum movement speed of the nodes, which ranges from 0-20m/s, each performance index under different maximum movement speeds is obtained, and the experimental results are shown in **Fig. 11** and **Fig. 12**.

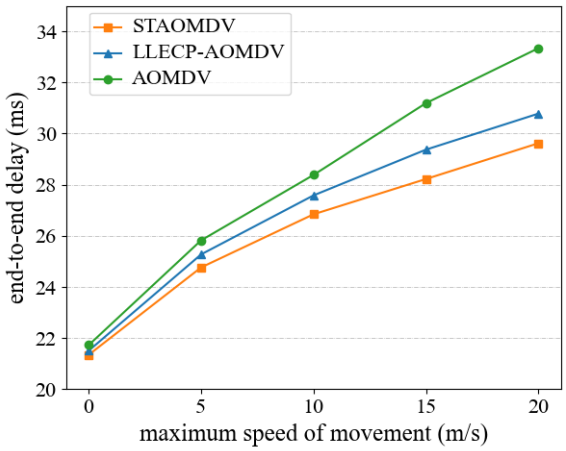


Fig. 11. Average end-to-end delay with different node maximum speeds

As can be seen from **Fig. 11**, the average end-to-end delay of STAMODV and the other two baseline schemes increases as the maximum node movement speed increases. As the node movement speed increases, the links between nodes are more likely to be lost due to the increased distance, requiring more frequent route discovery and maintenance, thus increasing the end-to-end delay. However, noteworthy observations from the figure reveal that STAMODV consistently maintains the smallest average end-to-end delay, whereas standard AOMDV exhibits the largest delay. This difference arises due to the fact that AOMDV solely considers the number of hops in path selection, while both STAMODV and LLECP-AOMDV account for connection losses due to device mobility and low residual energy of nodes, making the selected paths more stable and efficient. Furthermore, the presence of malicious nodes in the network results in an increased number of data retransmissions, leading to elevated data transmission delays. Notably, neither AOMDV nor LLECP-AOMDV considers this concern, while STAMODV addresses it effectively, resulting in improved end-to-end delay performance.

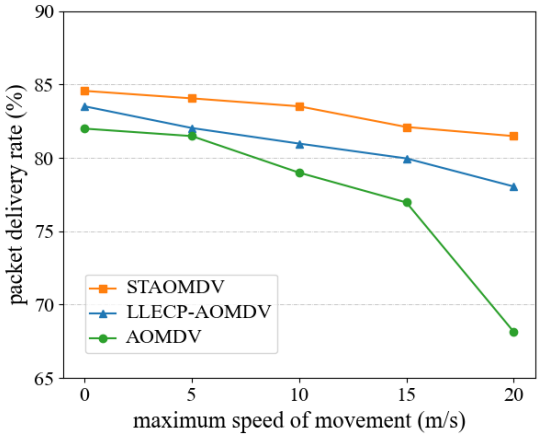


Fig. 12. Packet delivery rate with different node maximum speeds

The packet delivery rate performance is shown in **Fig. 12**, where the packet delivery rate of all three schemes decreases as the maximum speed of the node increases. As the node movement speed increases, the path selected during data forwarding may result in lower data

packet delivery rate due to link loss. However, thanks to the link lifecycle considered in link selection, the packet delivery rates of STAMODV and LLECP-AOMDV are significantly higher than those of AOMDV, and the decline is smaller. However, since LLECP-AOMDV does not consider the trust level of the paths, the selected paths may contain malicious nodes, resulting in a lower packet delivery rate than STAMODV in this paper.

5.2.2 Effect of malicious nodes

The maximum movement speed of the node is set to 10m/s. Simulation experiments are conducted by setting different numbers of malicious nodes (malicious behavior is assumed to be black hole attack), and each performance index is obtained as shown in **Fig. 13** and **Fig. 14**.

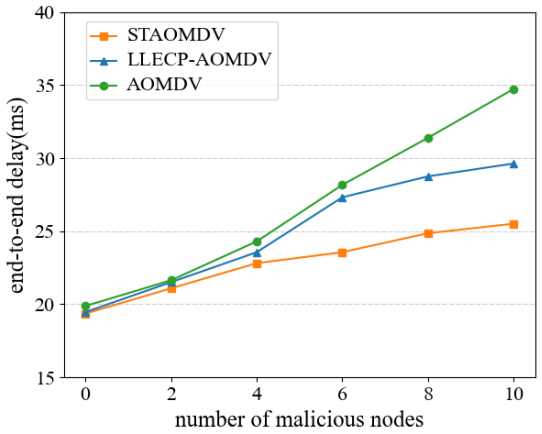


Fig. 13. The end-to-end delay with different numbers of malicious nodes

As shown in **Fig. 13**, the average end-to-end delay increases with the increase of the number of malicious nodes in the network. Due to the comprehensive consideration of the lifecycle, energy level and trust level of the link, the best path obtained may not be the shortest path, and the larger the number of malicious nodes. The larger the number of hops of its selected path may be, which makes the transmission delay increase; AOMDV only considers the number of hops when selecting a path for data transmission, and the path with the shortest number of hops may be unstable and insecure, and the probability of data transmission failure is positively correlated with number of malicious nodes is positively correlated, so the end-to-end delay increases more than STAMODV and LLECP-AOMDV.

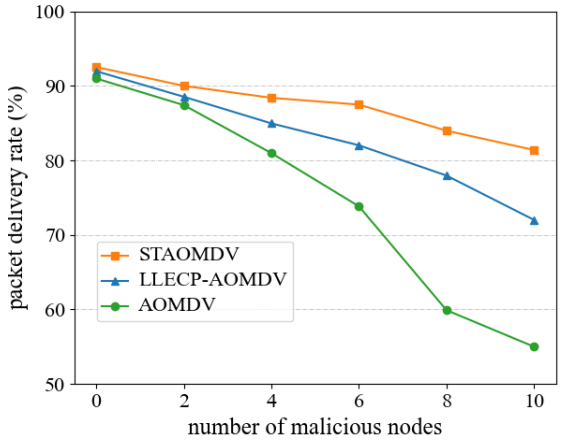


Fig. 14. Packet delivery rate with different numbers of malicious nodes

As shown in **Fig. 14**, the packet delivery rate decreases as the number of malicious nodes in the network increases. As malicious nodes launch attacks such as black hole attack, selective forwarding, etc., they maliciously discard the received data packets so that the data cannot reach the destination node, thus decreasing the packet delivery rate. However, as seen in **Fig. 14**, the packet delivery rate of STAMODV exhibits a more gradual decline compared to LLECP-AOMDV and AOMDV. This is due to the ability of STAMODV to determine whether a neighboring node is a malicious node by its forwarding behavior and thus select a more trusted path for data transmission, which the other two schemes do not achieve from hop count or link life cycle alone.

5.2.3 Energy balancing effect

In STAMODV, the energy level of the path is considered in path selection, and by selecting a path with higher energy level can balance the network energy consumption and extend the network lifetime. We fix the maximum node movement speed and the number of malicious nodes as 10m/s and 4, respectively, and observe the change of the number of energy-depleted nodes with the experimental simulation time, the results of which are shown in **Fig. 15**. From the experimental results, we can see that the number of energy-depleted nodes in both STAMODV and LLECP-AOMDV is small in the early stage of the experiment, and there is no node death in the first 100s, while there is one node death in AOMDV. At the later stage, the number of STAMODV and LLECP-AOMDV node deaths remained small, while AOMDV showed more node deaths. This is due to the fact that, both STAMODV and LLECP-AOMDV consider energy consumption during routing, which equalizes the energy consumption of nodes in the network and reduces the possibility of node death.

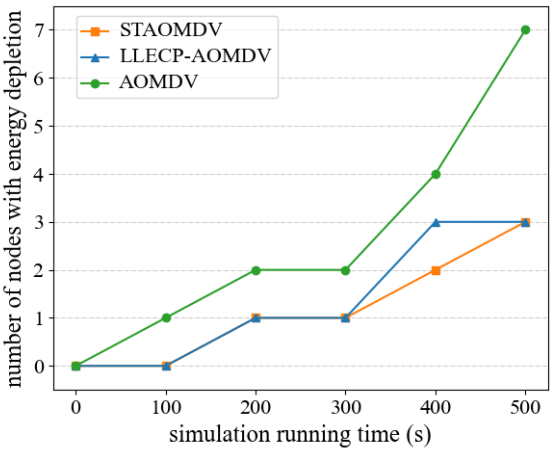


Fig. 15. Variation of the number of energy-depleted nodes with simulation time

5. Conclusion

In this paper, a trustworthy routing mechanism for the Internet of Things in edge computing scenarios was proposed in order to address the problem of multi-hop data transmission. Firstly, a trustworthy path selection strategy was designed based on three different link stability attributes, which comprehensively considered the link lifecycle, energy level, trust level, and identity authentication status to select the optimal path. Secondly, a trustworthy routing protocol based on link stability was proposed on top of the AOMDV protocol, which enabled

the establishment and maintenance of multi-path routing between nodes, as well as the collection and update of stability attribute data. Finally, simulation experiments were conducted to verify the performance of the STAOMDV protocol.

References

[1] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7457-7469, 2020. Article (CrossRef Link)

[2] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman and D. O. Wu, "Edge computing in industrial internet of things: Architecture, advances and challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462-2488, 2020. Article (CrossRef Link)

[3] L. Kong, J. Tan, J. Huang, G. Chen, S. Wang, X. Jin, P. Zeng, M. Khan and S. K. Das, "Edge-computing-driven internet of things: A survey," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1-41, 2022. Article (CrossRef Link)

[4] A. Rasheed, P. H. J. Chong, I. W. Ho, X. J. Li and W. Liu, "An overview of mobile edge computing: Architecture, technology and direction," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 10, pp. 4849-4864, 2019. Article (CrossRef Link)

[5] F. Meneghello, M. Calore, D. Zucchetto, M. Polese and A. Zanella, "IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182-8201, 2019. Article (CrossRef Link)

[6] D. Wu, S. Geng, X. Cai, G. Zhang and F. Xue, "A many-objective optimization WSN energy balance model," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 14, no. 2, pp. 514-537, 2020. Article (CrossRef Link)

[7] S. M. Muzammal, R. K. Murugesan and N. Z. Jhanjhi, "A comprehensive review on secure routing in internet of things: Mitigation methods and trust-based approaches," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4186-4210, 2021. Article (CrossRef Link)

[8] C. E. Perkins, P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *ACM SIGCOMM computer communication review*, vol. 24, no. 4, pp. 234-244, 1994. Article (CrossRef Link)

[9] D. B. Johnson, D. A. Maltz, J. Broch and Others, "DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks," *Ad hoc networking*, vol. 5, no. 1, pp. 139-172, 2001. Article (CrossRef Link)

[10] P. K. Maurya, G. Sharma, V. Sahu, A. Roberts, M. Srivastava, M. T. Scholar and Others, "An overview of AODV routing protocol," *International Journal of Modern Engineering Research (IJMER)*, vol. 2, no. 3, pp. 728-732, 2012. Article (CrossRef Link)

[11] L. A. Tawalbeh, F. Muheidat, M. Tawalbeh and M. Quwaider, "IoT Privacy and security: Challenges and solutions," *Applied Sciences*, vol. 10, no. 12, pp. 4102, 2020. Article (CrossRef Link)

[12] M. Karthigha, L. Latha and K. Sripriyan, "A comprehensive survey of routing attacks in wireless mobile ad hoc networks," in *Proc. of 2020 International Conference on Inventive Computation Technologies (ICICT)*, 2020. Article (CrossRef Link)

[13] V. Thirunavukkarasu, A. Senthil Kumar and P. Prakasam, "Cluster and angular based energy proficient trusted routing protocol for mobile ad-hoc network," *Peer-to-Peer Networking and Applications*, vol. 15, no. 5, pp. 2240-2252, 2022. Article (CrossRef Link)

[14] M. Hema Kumar, V. Mohanraj, Y. Suresh, J. Senthilkumar and G. Nagalalli, "Trust aware localized routing and class based dynamic block chain encryption scheme for improved security in WSN," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 5287-5295, 2021. Article (CrossRef Link)

[15] Z. Sun, M. Wei, Z. Zhang and G. Qu, "Secure routing protocol based on multi-objective ant-colony-optimization for wireless sensor networks," *Applied Soft Computing*, vol. 77, pp. 366-375, 2019. Article (CrossRef Link)

[16] F. S. Abkenar, P. Ramezani, S. Iranmanesh, S. Murali, D. Chulerttiyawong, X. Wan, A. Jamalipour and R. Raad, "A Survey on Mobility of Edge Computing Networks in IoT: State-of-the-Art, Architectures, and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2329-2365, 2022. Article (CrossRef Link)

[17] T. Khan, K. Singh, M. Manjul, M. N. Ahmad, A. M. Zain and A. Ahmadian, "A Temperature-Aware Trusted Routing Scheme for Sensor Networks: Security Approach," *Computers & Electrical Engineering*, vol. 98, pp. 107735, 2022. Article (CrossRef Link)

[18] B. Mathur, A. Jain, "AOMDV Protocol: A Literature Review," *International Journal of New Technology and Research*, vol. 4, no. 7, pp. 27-30, 2018. Article (CrossRef Link)

[19] N. V. Brindha, V. S. Meenakshi, "A secured optimised AOMDV routing protocol in MANET using lightweight continuous multimodal biometric authentication," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-17, 2022. Article (CrossRef Link)

[20] Y. Yao, B. Xiao, W. Wang, G. Yang, X. Zhou and Z. Peng, "Real-time cache-aided route planning based on mobile edge computing," *IEEE Wireless Communications*, vol. 27, no. 5, pp. 155-161, 2020. Article (CrossRef Link)

[21] T. Wang, Y. Liang, X. Shen, X. Zheng, A. Mahmood and Q. Z. Sheng, "Edge Computing and Sensor-Cloud: Overview, Solutions, and Directions," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1-37, 2023. Article (CrossRef Link)

[22] X. Chen, Y. Bi, X. Chen, H. Zhao, N. Cheng, F. Li and W. Cheng, "Dynamic Service Migration and Request Routing for Microservice in Multicell Mobile-Edge Computing," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13126-13143, 2022. Article (CrossRef Link)

[23] H. P. Mistry, N. H. Mistry, "RSSI based localization scheme in wireless sensor networks: A survey," in *Proc. of 2015 Fifth International Conference on Advanced Computing & Communication Technologies*, 2015. Article (CrossRef Link)

[24] S. Zhang, D. Cao and Z. Ning, "A decentralized and reliable trust measurement for edge computing enabled Internet of Things," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 24, pp. e7238, 2022. Article (CrossRef Link)

[25] H. Xia, J. Yu, C. Tian, Z. Pan and E. Sha, "Light-weight trust-enhanced on-demand multi-path routing in mobile ad hoc networks," *Journal of Network and Computer Applications*, vol. 62, pp. 112-127, 2016. Article (CrossRef Link)

[26] M. G. R. Alam, M. M. Hassan, M. Z. Uddin, A. Almogren and G. Fortino, "Autonomic computation offloading in mobile edge for IoT applications," *Future Generation Computer Systems*, vol. 90, pp. 149-157, 2019. Article (CrossRef Link)

[27] S. Mueller, R. P. Tsang and D. Ghosal, "Multipath routing in mobile ad hoc networks: Issues and challenges," in *Proc. of MASCOTS 2003: Performance Tools and Applications to Networked Systems: Revised Tutorial Lectures*, pp. 209-234, 2004. Article (CrossRef Link)

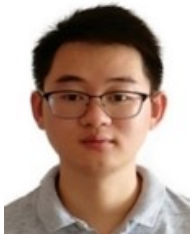
[28] B. Djamaa, M. R. Senouci, H. Bessas, B. Dahmane and A. Mellouk, "Efficient and stateless P2P routing mechanisms for the Internet of Things," *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11400-11414, 2021. Article (CrossRef Link)

[29] D. Zhang, L. Chen, J. Zhang, J. Chen, T. Zhang, Y. Tang and J. Qiu, "A multi-path routing protocol based on link lifetime and energy consumption prediction for mobile edge computing," *IEEE Access*, vol. 8, pp. 69058-69071, 2020. Article (CrossRef Link)

Dongzhi Cao is a Ph.D. candidate of Faculty of Information Technology, Beijing University of Technology, China. She received her M.S. degree in Computer Technology from Beijing University of Technology, China, in 2020. Her main research interests include trusted computing, edge computing and network security.

Peng Liang is a senior engineer at the China Center for International Economic Exchanges. He received the Ph.D. degree in Computer Science and Technology from Beijing University of Technology, China, in 2014. His main research interests include information security and digital economy.

Tongjuan Wu is an engineer at the Beijing HIWING Scientific and Technological Information Institute. She received the M.S. degree in Industrial Design Engineering from Peking University, China, in 2019. Her main research interests include intelligent interaction and intelligent optimization.



Shiqiang Zhang is a Lecturer at the School of Information, Beijing Wuzi University. He received his B.S. degree (2016) in Computer Science and Technology from Beijing University of Chemical Technology, China. He received the Ph.D. degree in Computer Science and Technology from Beijing University of Technology, China in 2021. His main research interests include edge computing, network security, and big data analysis.

Zhenhu Ning is a Lecturer at the Faculty of Information Technology, Beijing University of Technology. He received the Ph.D. degree in Computer Science and Technology from Beijing University of Technology, China, in 2016. His main research interests include IoT security, cloud security, malicious code detection, machine learning, system optimization and control, and practical partial differential equations.